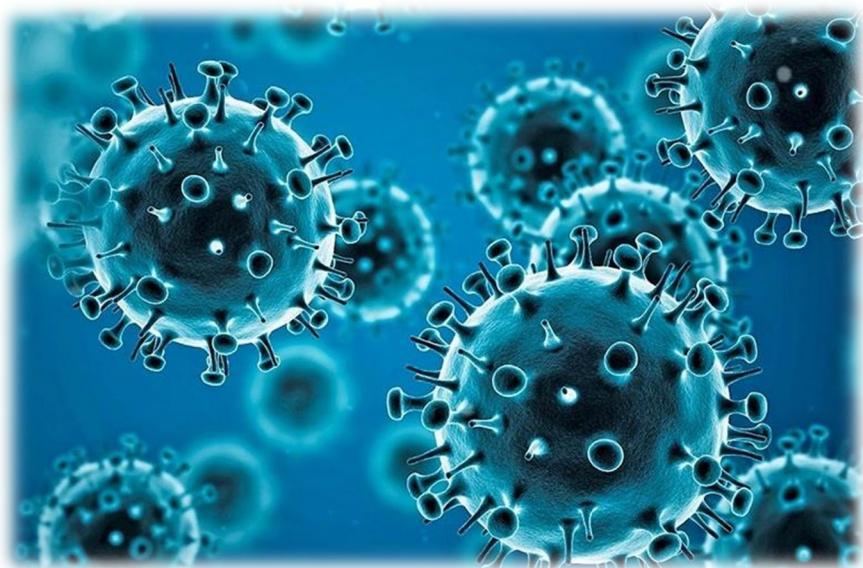


חיזוי מספר החוליםים בקורונה בישובים בישראל



מגישיים :

- אייל לוטן, ת"ז : 302288527. מייל : eyallotan@campus.technion.ac.il
- דור סורה, ת"ז : 204098784. מייל : dorsura@campus.technion.ac.il
- תומר הבר, ת"ז : 307916502. מייל : tomer.haber@campus.technion.ac.il

מנחה : מר רפאל גד

מרצה אחראי : פרופ' שאול מרכוביץ'

תוכן עניינים

4.....	הקדמה
5.....	Data
5.....	Feature Extraction
5.....	טבלאות
6.....	Preprocessing
10.....	Train & Test Sets
11.....	מטריקות המדידה
15.....	אלגוריתמי חיזוי קלאסיים
15.....	KNN
16.....	KNN Regression
18.....	מתודולוגיה ניסויית
19.....	אימון האלגוריתם
22.....	Feature Selection
23.....	אימון K
23.....	בחירה פונקציית משקל
24.....	תוצאות וסיכום
27.....	Decision Tree
27.....	רָקֵע תאורטי
31.....	חלק א' – עץ החלטה אל מול עיר החלטה
33.....	חלק ב' – בחירת תכונות רלוונטיות ביותר
39.....	חלק ג' – החלפת תכונות קיימות בתכונות חדשות
41.....	סיכום
42.....	Time Series Analysis
42.....	רָקֵע תיאורטי
42.....	תכונות מתמטיות וניתוח סטטיסטי
45.....	Statistical Hypothesis Testing
48.....	Time Series Transformations
55.....	Time Series Forecasting
55.....	Data Preprocessing
57.....	Moving Average Model
65.....	Auto Regression Model
71.....	ARMA Model
75.....	ARIMA Model
79.....	SARIMA Model
86.....	השוואת מודלים
87.....	סיכום
88.....	הדגמות וניסויים
88.....	Auto ARIMA

95	Rolling Forecast
99	Rolling Average Smoothing
105	סיכום
106	סיכום הפרויקט
106	דיון בתוצאות
107	מה חסר במה שעשינו?
108	כיוונים להמשך מחקר
110	הוראות הרצה
111	ביבליוגרפיה

הקדמה

מגפת הקורונה (COVID-19) היא מגפה עולמית אשר פוקדת את העולם מאז התגלתה לראשונה בראשונה בדצמבר 2019 בעיר ווהאן שבסין. המחלת התפשטה במהירות ברחבי העולם והובילה למקרי הדבקה רבים. בחודש מרץ 2020 ארגון הבריאות העולמי הגדר באופן رسمي את מחלת הקורונה כמגפה עולמית. נכון לכתיבת שורות אלו תועדו יותר מ-290 מיליון מקרי הדבקה במחלת, וכ-5.5 מיליון בני אדם נפטרו מהמחלה וסיבוכיה. בכך הפכה מגפת הקורונה להיות אחת מהמגפות הקטלניות ביותר ביוטר אשר פקדו את המין האנושי.

בחודש פברואר 2020 התגלו לראשונה החולים בישראל. מדינת ישראל, כמו שאר מדינות העולם, מתמודדת עם התפרצויות המחלת והשלכותיה, וקבעה הנחיות למניעת הדבקה בנגיף ולהאטת התפשטותו. הعليיה בתחום ההובלה ממשלות ורבות ברחבי העולם להטיל סנקציות על התושבים, החל מבידוד ביתני לאנשים שנחשפו לחולה לאומיות, ועד לסתורים על כלל האוכלוסייה הכלולים איסור יציאה מוחלט מהבית (למעט לצרכים חיוניים).

התפשטות המגפה בארץ ובעולם לוותה במספר כלכלי משמעותי. מקומות העבודה רבים נאלצו לפטר עובדים או להוציאם לחיל"ת (חופשם ללא תשלום), וענפי מסחר רבים ספגו מכחה כלכלית אנושה כתוצאה מהנסיבות הרוחיק החברתי והסגרים השונים.

אחד הביעות המרכזיות עימה מתמודדות הממשלה ומערכת הבריאות בארץ, היא יכולת לחזות את התפרצויות המגפה. חיזוי מדויק של התפרצויות תחולאה באזורי מסויים יכולה לסייע במגוון דרכים:

1. מתן מענה ממוקד ונקיית אמצעים שיוibiliו להקלת של התפרצויות לפני שהיא יוצאת משליטה.
2. הכנה של מערכת הבריאות ובתי חולים לקליטת החולים.
3. העברת עובדים לא חינויים לעבודה מהבית.
4. העברת בתים ספר למשרדים מרוחקים.
5. העלאת כמות הבדיקות המבוצעות ליום, ועידוד של אזרחים להתחסן ו/או לבצע בדיקות ביתיות ולא לצאת מהבית במידה והם חשים ברע.

כל הצעדים שהוזכרו לעיל (בנוסף לעוד צעדים רבים נוספים שלא הוזכרו) יכולים לעזור בצורה משמעותית למניעת הדבקה המונית ותחולאה קשה, וכתוכאה מכך לצמצם את הפגיעה של המגפה בכל תחומי החיים.

כאשר המגפה פרצה לראשונה, מדיניות הממשלה הייתה אגרסיבית מאד וכללה סגרים הרטתיים מtooך מטרה למגר את התחולאה לגמרי. לאחר שנתיים של התמודדות עם המגפה, הממשלה הבינה כי עליה לגבות תוכניות שתאפשר "חיים לצד הקורונה", מtooך הבנה שהmagפה לא עתידה להיעלם לחוטין בעתיד הקרוב. על מנת לגבות מדיניות שכזו, יש צורך בהתאם מתמדת של הנהלים וההנחיות בהתאם למצב התחולאה הנוכחי. חיזוי טוב של מגמות התחולאה והתפרצויות המגפה יוכל לעזור לממשלה להתאים את ההנחיות שלה במהרה ובכך להיות תמיד אחד אחד לפני המגפה.

הבעיה אותה נרצה לפתור בפרויקט זה היא זיהוי גרי התחולאה בكورونا על בסיס נתונים של תחולאות הקורונה בעיר ישראל ומרכיבים נוספים אשר משפיעים על התחולאה. מטרתנו תהיה למדל את הבעיה במספר דרכים שונות, למצוא את התכונות הרלוונטיות שייעזרו בניבוי התחולאה, ולהציג אלגוריתמים ואופטימיזציות שונות אשר יובילו אותנו להשגת חיזוי מיטבי.

Data

באטר ממשרד הבריאות מתעדכנים מידי יום מספר רב של datasets באשר לנוטוני הקורונה במדינת ישראל. ניתן למצוא שם נתונים רבים על מספרי נדבקים, מחוסנים, נפטרים ועוד. המידע בניוי כך שבכל יום, עבור כל יישוב, נוספה שורה עם המידע המעודכן לאותו היום.

בפרויקט זה נעשה שימוש נתונים של מספרי החולים ביישובים מפוץ הקורונה בישראל ועד לתאריך ה-22/11/2021. בכל פרק נבצע חיזוי של תקופות התפרצויות שונות.

Feature Extraction

בחלק הראשון של הפרויקט נטמקד בפתרון בעיית החיזוי באמצעות שימוש באלגוריתמי למידה קלאסיים. על מנת לעבוד עם אלגוריתמים אלו, علينا ליצור dataset המכיל וקטוריים של תוכנות אשר ישמשו את המודלים ללמידה, ועל בסיסים ייעשה לאחר מכן החיזוי.

שלב בנית התוכנות הוא שלב חשוב מאוד לבניית מודלים בתחום הבינה המלאכותית. ה-data שלנו הוא גולמי ולא בניו באופן בו אלגוריתמי הלמידה שלנו יוכל לעשות בו שימוש. בשלב בנית התוכנות, נרצה ליצור וקטור של תוכנות לכל יישוב בכל יום על מנת לחזות את מספר החולים בו.

חקרוו מספר רב של טבלאות וחיפשנו אילו נתונים יתנו את האינדייקציה הטובה ביותר לחיזוי מספר החולים היומי. הטבלאות בנויות כך שבכל שורה יש דאטה מצטבר עד אותו היום.

טבלאות

נציג את הטבלאות בהן השתמשנו על מנת ליצור את ה-dataset שלנו :

- **נתונים ברמת היישוב :**

קוד יישוב	שם היישוב	תאריך	מספר מאומתים	מספר מחלימים	מספר נפטרים	מספר בדיקות	ציון הרמה	צבע הרמה
קוד 2-4 ספרות רק לשיכונים הגדולים מ-2000 תשכום, נכון לסופ' 2019	שם היישוב רק של יישובים הגדולים מ-2000 תשכום, נכון לסופ' 2019	31.8.2020	31.8.2020, החל מה- 31.8.2020					

• **נתוני ההתחסנות לפי יישוב:**

תאריך										קוד יישוב									
שם היישוב					סמלים					תקן יישוב					תקן מגורים				
סמלים		תקן יישוב			תקן מגורים		תקן יישוב			תקן מגורים		תקן יישוב			תקן מגורים		תקן יישוב		
- 1 + 90 גיל	מינה - 1 גיל 80- 89	מינה - 1 גיל 70-79	מינה - 1 גיל 60-69	מינה - 1 גיל 50-59	מינה - 1 גיל 40-49	מינה - 1 גיל 30-39	מינה - 1 גיל 20-29	מינה - 1 גיל 0-19	שם היישוב	סמלים	תקן יישוב	תקן מגורים	תקן יישוב						
אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	המחלם - 20.12.2021	תקן מגורים הגולים מ-2000- תשובים, נכון ל-ט'ו' שנת 2019	תקן יישוב רץ' אנו רץ' מספר	תקן מגורים רץ' אנו רץ' מספר							
מינה - 2 גיל + 90	מינה - 2 גיל 80- 89	מינה - 2 גיל 70-79	מינה - 2 גיל 60-69	מינה - 2 גיל 50-59	מינה - 2 גיל 40-49	מינה - 2 גיל 30-39	מינה - 2 גיל 20-29	מינה - 2 גיל 0-19	מינה - 3 גיל + 90	מינה - 3 גיל 80- 89	מינה - 3 גיל 70-79	מינה - 3 גיל 60-69	מינה - 3 גיל 50-59	מינה - 3 גיל 40-49	מינה - 3 גיל 30-39	מינה - 3 גיל 20-29	מינה - 3 גיל 0-19	מינה - 3 גיל + 90	
אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	אנו רץ' מספר	המחלם - 20.12.2021	תקן מגורים הגולים מ-2000- תשובים, נכון ל-ט'ו' שנת 2019	תקן יישוב רץ' אנו רץ' מספר	תקן מגורים רץ' אנו רץ' מספר							

לאור ניסיון העבר עם נגיף קורונה, אנו מעריכים שמספר המאומתים היומי תלוי במספר המאומתים ביום שקדמו לו. לכן נרצה לבנות תכונות של מספרי החולים ביום שקדמו ליום הנוכחי. כל ישוב בכל יום נרצה לחשב את אחוזו חוליו הקורונה החדש בכל אחד מהימים האחרונים.

בנוסף, אנחנו מעריכים שהחיסונים משפיעים מאוד על ההדבקה ועל מספר המאומתים לנגיף הקורונה. לכן נרצה להוציא לפחות יומי אחד מתחסנים חדשים שחיו בשבועות האחרונים כדי לבטא את רמת החיסוניות בישוב (כיצענו ארגזציה לאחוזה מתחסנים כולל ולא לפיקניילאים כפי שמוצג בטבלה לעיל).

Preprocessing

בשלב ה-Preprocess אנחנו עושים ל-data הגולמי שלנו טרנספורמציה לבנייה טבלאי המתאים לשימוש באלגוריתמי הלמידה שלנו. תהליך זה כולל התמודדות עם בעיות שונות במבנה ה-data, ייצור תכונות חדשות מהתכונות המקוריות בהתאם לנדרש ונורמל ערכיהם.

נתאר את השלבים השונים שביצעו בשלב ה-**preprocessing**:

- ראשית היינו צריכים להתמודד עם אילוץ של **data** אליו עבדנו – כאשר תוכונה מסוימת (כגון מספרי החולמים או המחוסנים) יהיה נזוק, הערך עבור אותה תוכנה מכיל את המחרוזת "**15**" שאומר שיש פחות מ-15 מקרים עבור התוכנה הרלוונטית (לדוגמא פחות מ-15 חולמים הוגלו בישוב מסוימים).

בהתחלת ניסינו לשים בכל המיקומות האלה את הערך 0 אבל לאחר תחילת העבודה הבנו כי החלפה זו היא גסה מדי ומיצרת עבורנו data המלא בעברכי 0 אשר מקשים מאד על אלגוריתמי הלמידה לבצע חיזוי אמין. רצינו לבצע הפרדה יותר עדינה עבור הערכים הביניים הללו. כדי למלא את השורות האלה ב-data שימושי, לקחנו עבור כל ישוב את כל טווח התאריכים בו מספר חולים/מחוסנים (או כל תכונה אחרת רלוונטי) היה קטן מ-15, וחילקנו אותו ל-15 תחומים (בכל

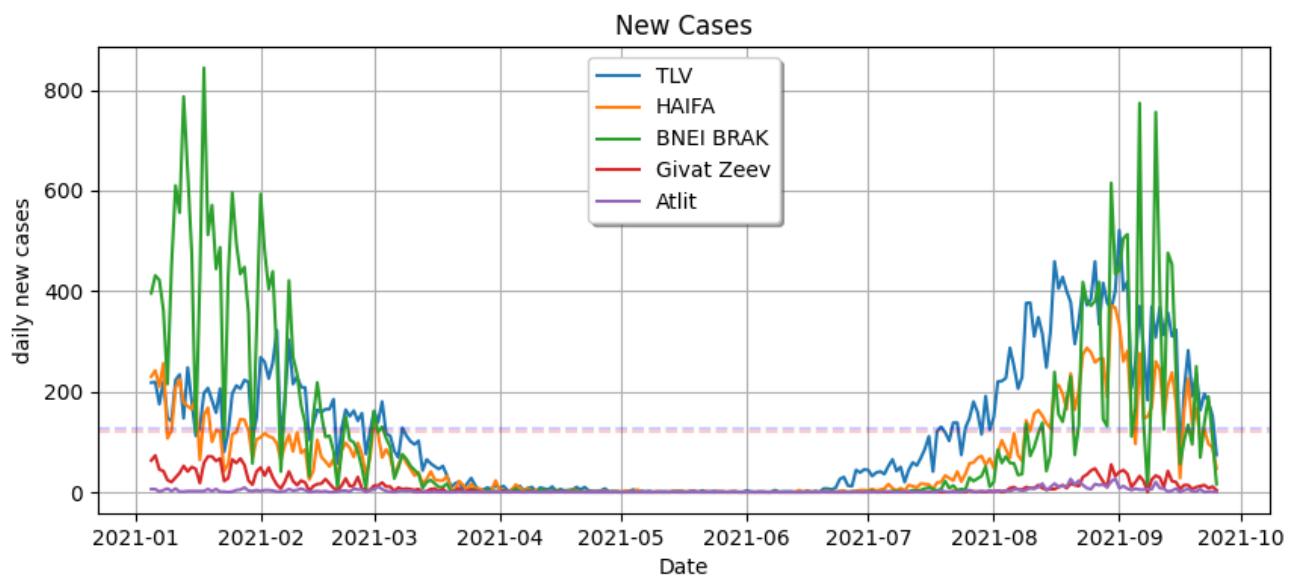
תחום יכול להיות שורה אחת, יותר או פחות) ובכל תחום שתלנו מספר עלה מ-1 ועד 14. באופן זה מילאנו את data-ו שלנו בצורה שווה, ובנוספ' אנחנו מאמינים כי פעולה זו לא פוגעת באמינות הנתונים מכיוון שביצענו הנחה סבירה כי הערכים הנומכים מתפלגים בצורה אחידה על פני טווח התאריכים הרלוונטי.

- תוכנת צבע הרמזור הכליה מחרזות עם הצבע הרלוונטי (ירוק/צהוב/כתום/אדום). על מנת להפוך את הנתונים שלנו למספריים המרנו את הצבעים במספרים בטוויה 3-0 (כאשר 0 מייצג צבע ירוק ו-3 מייצג צבע אדום). בנוסף, בחרנו תאריך תחילת נתונים dataset- שלנו בו כבר התחילו להשתמשocabularies.
- כדי לחשב את מספר החולים היומי ביצענו את התהליך הבא: ביצענו shift לעמודה המכילה את מס' המאומתים המצטבר (Cumulative_verified_cases) כך שכל העמודה "ירדה" שורה אחת למיטה", ואז החסרנו את מס' המאומתים המצטבר עם העמודה החדשה ובכך קיבלנו את מספר החולים באותו יום. ביצענו את התהליך הזה N פעמים כדי לייצר את מספר החולים היומי N ימים אחריה. עבור ה dataset- שלנו קבענו את הערך N להיות שווה ל-14, כלומר כל וקטור מכיל את הנתונים על החולים ב-14 ימים שקדמו ליום הנוכחי. על מנת לא להתמודד עם מקרי קצה, דאגנו לחתך תאריך תחילת נתונים dataset- שלנו בו כבר קיימים נתונים עבור 14 ימים אחרת.
- עמודות החיסונים מציגות עבור כל יום, את מספר המוחסנים בכל אחת ממנות החיסון (מס' המנות הרלוונטיות עבור תקופה התאריכים אותה אנחנו עובדים בפרויקט זה היא שלוש מנות) בשבוע ובשבועיים האחרונים הקודמים לתאריך הנוכחי. כדי לחשב עמודה זו ביצענו תהליך דומה לזה שבוצע עבור חישוב מס' החולים ביום הקודם, למעט העבודה שסמכנו את הערכים עבור כל אינטראול רלוונטי (שבוע/שבועיים). את הנתונים לקחנו מטבלת נתוני התחשנות שהוצאה לעיל. גם כאן נאלצנו לבחור תאריך תחילת נתונים dataset- שלנו בו כבר התחילו להציג נתונים התחשנות (כלומר החל מהתחלת החיסון הראשונית).
- נרמול עמודות – ברצוננו להשתמש **בחזויות** עבור חלק מהתכונות, במקום **במספר האבסולוטי**, וזאת בכך ש data- של יישוב מסוים יעוזר ליישוב אחר. לדוגמה, אם עבור יום מסוים ישנים 100 החולים בתל אביב או 100 החולים בעתלית כਮובן שזה לא מייצג את אותה חומרת מצב. אבל אם 0.01% החולים חולו בתל אביב ו-0.01% חולו בעתלית כמובן שזה תואם יותר. כך וקטור האימון של תל אביב יוכל לעזור לקטור אימון של עתלית ולהפוך. כדי לעبور ל- data- המבוסס על אחוזים, علينا לנормל את כל העמודות הרלוונטיות המבוססות על מספרים אבסולוטיים (כגון מספר החולים/מחוסנים/נפטרים וכו'). לצורך ביצוע הנרמול, השתמשנו בטבלה חיצונית המכילה את מספרי התושבים בכל יישוב. על מנת להשלים את הנרמול, חילקנו את הערכים של כל עמודה הרלוונטית לנרמול במספר התושבים ביישוב המתאים (כל שורה נורמלה בהתאם למספר התושבים בעיר שמייצגת השורה). נציין כי ביצענו ניסיונות להכפיל את הערכים המנורמלים שקיבלו בקבוע כלשהו (ובכך לקבל את אחוז החולים/מתחסנים/נפטרים ל-10,000 איש או ל-100,000 איש), אך ניסויים שביצענו הראו כי אין益处 בהפלה זו.
- כדי לקבל אחידות מבחינות נרמול הנתונים, ביצענו נרמול גם עבור שאר העמודות הרלוונטיות (ציון רמזור, צבע רמזו).

נשים לב שהנرמול נתן לנו תוכנה משמעותית.icut, נקבל שככל התוכנות שלנו מנורמלות לטוויה שבין 0 עד 1, וכך פונקציות המרחק של האלגוריתמים מתייחסות לכל התוכנות באופן שווה.

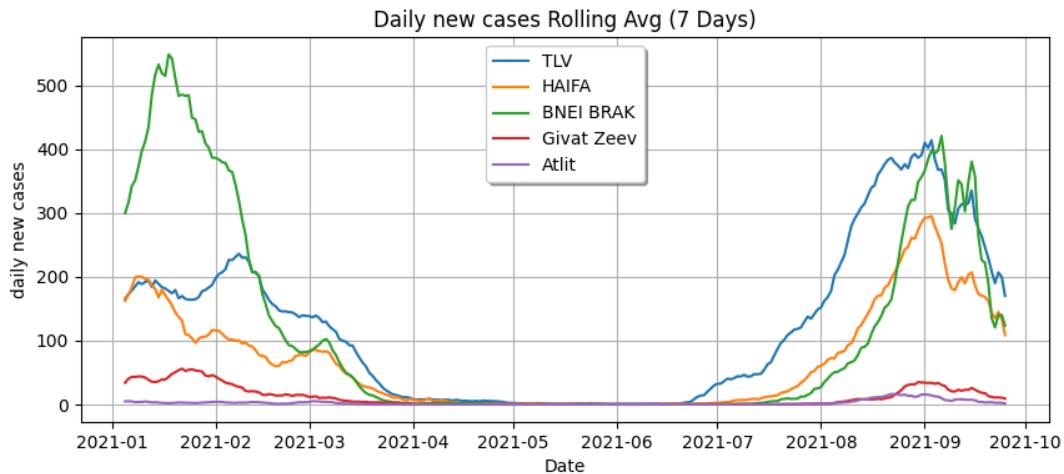
ניתן לראות כי על מנת להתאים את ה-dataset שלנו לbehor תאריך התחלת מאוחר יותר עבור דיווח הנתונים בטבלה, וזאת על מנת שכל התוכנות שאוthon נרצה למدل יהו רלוונטיות ולא יכילו ערכים ריקים. כדי לbehor את התאריך ההתחלה הסופי עבור הנתונים שלנו, ניקח את התאריך המקסימלי מכל האילוצים שהוצעו לעיל. עבור הדוגמאות המוצגות כאן נקבע את טווח התאריכים של ה-dataset שלנו להיות החל מהתאריך ה-25/09/2021 ועד ל-05/01/2021.

מציג את גרפ המאומתים היומי עבור מספר יישובים נבחרים :



ניתן לראות שב-data שלנו יש הרבה "רעש", וכי הפונקציות אותן אנחנו מנסים לחזות לא "חלקות". בחלק העוסק ב-Time Series Analysis נסביר באופן פורמלי מהי משמעות ה-"רעש" הקדים לנו נתונים, אך מבחינה אינטואיטיבית ניתן להבין כי מדובר בחוסר יציבות בערכים הנמדדים עבור התמונה אותה אנחנו מנסים לבנה. ההשערה שלנו היא שהוסר יציבות זה נובע מהבדלים בין ימי חול לימי סוף השבוע. בימי שישי ושבת (כמו גם בחגים ומועדים), ישנו פחות נבדקים ולכן מספר החוליםים היומי נמוך יותר בהתאם. כל זאת למרות שאחוז החוליםים היומי לא בהכרח משתנה. כפי שנראה בפרק הבא, ההתמודדות עם הנתונים בתצורה הזאת יכול להקשות על החיזוי.

בחלק הניסיוני להתמודד עם הקושי שהוצע לעיל, נוכל להציג טרנספורמציה מהבעיה המקורית שלנו לבעיית חיזוי חדשה (אך דומה). במקומות לחזות את כמות החוליםים עבור יום מסויים, נוכל לנסות ולהזות את ממוצע החוליםים בשבועת הימים האחרונים (כולל היום הנוכחי). הסבר נוסף על טרנספורמציה זו והאינטואיציה לבחירתה יובא בפרק Time Series Analysisrolling Average Smoothing. חיזוי של הבעיה החדשה אמנם אינו שקול לחיזוי של הבעיה המקורית אותה אנחנו מנסים לפטור, אך ניתן להסתכל על בעיה זו ועל "קרובות משפחה" של הבעיה המקורית שלנו. הסיבה לטרנספורמציה זו היא שכעת קיבל פונקציה אשר תהיה קלה יותר לחיזוי. לצורך הדוגמה, מציג את הגрафים עבור אותן הערים שהצגנו לעיל, אך כתע נחשב עבורן את פונקציית rolling average עבור תקופה של 7 ימים:



ניתן לראות כי כתת הגרפים הרבים יותר "חלקים" ואין תנודתיות גבוהה כפי שקיבלונו עבור התוכונה המקורית שלנו. בפרקם הבאים נשתמש בטרנספורמציה זו על מנת להציג את ההבדלים בין חיזויי הבעה המקורית לחיזוי הממוצע הנע.

נתאר את רשימת התכונות הסופית שלנו:

- City_Name – שם היישוב – עמודה זו לא שימשה בזמן הלמידה.
- City_Code – מזהה היישוב.
- Date – התאריך הרלוונטי עבור הרשומה.
- Colour – זהה עמודה שmbטאת את צבע היישוב (ביום שקדם לו). 0 – ירוק, 1/3 – צהוב, 2/3 – כתום, 1 – אדום.
- Final_score – ציון הרمزורי (בין 1-10, מנורמל).
- Vaccinated_dose_1_total – אחוז המתחסנים במנה 1 ביישוב.
- Vaccinated_dose_2_total – אחוז מתחסנים במנה 2 ביישוב.
- Vaccinated_dose_3_total – אחוז מתחסנים במנה 3 ביישוב.
- dose_1_in_last_1_week – אחוז האנשים שקיבלו מנתה ראשונה בשבוע האחרון.
- dose_1_in_last_2_week – אחוז האנשים שקיבלו מנתה ראשונה בשבועיים האחרונים.
- dose_2_in_last_1_week – אחוז האנשים שקיבלו מנתה שנייה בשבוע האחרון.
- dose_2_in_last_2_week – אחוז האנשים שקיבלו מנתה שנייה בשבועיים האחרונים.
- dose_3_in_last_1_week – אחוז האנשים שקיבלו מנתה שלישית בשבוע האחרון.
- dose_3_in_last_2_week – אחוז האנשים שקיבלו מנתה שלישית בשבועיים האחרונים.
- today_verified_cases – מספר החוליםים היומי – זהו הערך אותו אנחנו רוצחים לחזות (Y_True).
- rolling_average_7_days – זהה עמודת Y_True החלופית (עבור הטרנספורמציה לביעית ה-rolling average).
- verified_cases_i_days_ago – אחוז החוליםים היומי ביישוב לפני i ימים $i \in \{1, \dots, 14\}$.

- אחוז החולמים המציגים עד יום לפני התאריך הנוכחי. Cumulative_verified_cases •
- אחוז המחלימים המציגים עד יום לפני התאריך הנוכחי. Cumulated_recovered •
- אחוז המתים המציגים עד יום לפני התאריך הנוכחי. Cumulated_deaths •
- אחוז הבדיקות בישוב עד יום לפני התאריך הנוכחי. Cumulated_number_of_tests •
- אחוז הבדיקות לאיתור מאומתים בישוב עד יום לפני התאריך הנוכחי. Cumulated_number_of_diagnostic_tests •

Train & Test Sets

חלוקת ה-data ל-train ו-test מושגת על ידי שימוש בתאריך הפרדה. משום שהוא מתקשר בבעיית time-series, ה-test חייב להיות אחרי ה-train (כי לא ניתן לחזות רשותות "מה עבר" באמצעות רשותות "מהעתיד"). לכן בחרנו תאריך הפרדה כאשר כל הרשותות לפני הן רשומות train, וכל הרשותות אחרי הן רשומות test.

מטריקות המדידה

בפרויקט זה נעשה שימוש במספר מטריקות מדידה על מנת להעריך את טיב האלגוריתמי הלמידה בהם השתמשנו. לצורך הצגת המטריקות נגדיר מספר פרמטרים :

N – Number of examples, y_{true_i} – i_{th} y true value, y_{pred_i} – i_{th} y predicted value

כעת נציג את המטריקות המדוברות ונסביר מהי המוטיבציה העומדת מאחוריה השימוש בהן.

• : **MAE – Mean Absolute Error** •

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_{true_i} - y_{pred_i}|$$

שימוש במטריקה זו מודוד את השגיאה האבסולוטית המצטברת וזאת בהסתכלות על ממוצע כלל השגיאות. מטריקה זו מועילה לנו בכך שנוכל לדעת עד כמה ערכי החיזוי קרובים לערכי האמת אותן אנו מנסים לחזות.

• : **MSE – Mean Squared Error** •

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_{true_i} - y_{pred_i})^2$$

• : **RMSE – Root Mean Squared Error** •

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_{true_i} - y_{pred_i})^2}$$

שתי המטריקות (MSE ו-RMSE) דומות במשמעותם למטריקה MAE רק שכעת ב- MSE אנו סוכמים את ריבועי השגיאות ומחלקים בסך הדוגמאות ובמטריקה $RMSE$ מתבצע חישוב נוסף שהוא שורש של התוצאה המתקבלת.

לאחר הצגת שלוש המטריקות נציג את הדוגמא הבאה. נניח שברצוננו לחזות את מספר חוליות הקורונה ביישוב מסוים ביום ספציפי. נשים לב שעבור 1 , $y_{true} = 10$, $y_{pred} = 10$ נקבל שגיאה אבסולוטית השווה ל- 9 ועבור 101 , $y_{true} = 110$, $y_{pred} = 101$ נקבל שוב את אותה השגיאה. נשים לב שהקשר של חיזוי מספר חוליות קורונה, קיים הבדל מהותי בין התוצאות שהרי התוצאה הראשונה תנבע לנו חוליה אחד מה שיוביל לאבחנה שההדבקה ביישוב המדובר נמוכה פי עשרה מההדבקה האמיתית (מציאות אשר עלולה לגרום להתעלמות מההפרצות המחללה באותו יישוב), ואילו בדוגמה השנייה נקבל ערך אשר ייעיד עבורנו על מספר חולים אשר איננו שונה במעט מהערך האמתי באותו היישוב שהרי אף עבור 101 חולים נרצה לנסות ולמגר את מקדם ההדבקה. לכן נציג מטריקה אשר תעריך את **אחוז השגיאה** ובכך תעיד עבורנו על שונות בין הדוגמאות לעיל.

: **MAPE – Mean Absolute Percentage Error** •

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|y_{true_i} - y_{pred_i}|}{y_{true_i}}$$

מטריקה זו מייצגת את אחוז השגיאה הממוצע על סך כל הדוגמאות. נחזור לדוגמא שלעיל. ב庆幸 באמצעות מטריקה זו, קיבל שכאשר $y_{true} = 10, y_{pred} = 1$ אחוז השגיאה המתקבל יעמוד על 0.9 ואילו עבור הדוגמא בה $y_{true} = 110, y_{pred} = 101$ נתקבל אחוז שגיאה נמוך באופן משמעותי השווה ל-0.081. מטריקה זו תהיה שימושית מאד עבורנו על מנת להעריך את טיב החיזוי. חשוב לציין כי ישנה בעיות שימוש במטריקה זו במקרים בהם $y_{true_i} = 0$. ספריית המטריקות בה השתמש מטפלת בבעיה זו על ידי הגדולה של הפונקציה $\max(\epsilon, y_{true_i})$ באופן הבא:

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|y_{true_i} - y_{pred_i}|}{\max\{\epsilon, y_{true_i}\}}$$

כאשר $\epsilon < 0$ הוא קבוע חיובי קטן מאד. נשים לב כי במקרים בהם נתעסק בחיזוי של אזורים בהם התחלואה נמוכה (באזורים בהם $y_{true_i} = 0$ עבור יום כלשהו), המטריקה זו עלולה "להתפוץץ" ולתת ערכים מאד גבוהים. במקרים אלו ניאלץ להעריך את טיב החיזוי באמצעות אמצעים אחרים.

R – squared Score •: במטריקה זו נעשה שימוש נרחב בפרק הפרויקט. נציג כיצד פונקציה זו מחושבת. לצורך פשוטות השתמש במטרה הפרויקט שלנו, לחזות את מספר המאומתים ביישוב ספציפי ביום מסוים. תהיו קבוצה המכילה ערכי דוגמאות $\{y_n, \dots, y_1\}$ המייצגים את המספר האמיתי של חיובים לקורונה ביום ספציפי. נסמן ב- \bar{y} את ממוצע ערכי קבוצה זו. כלומר:

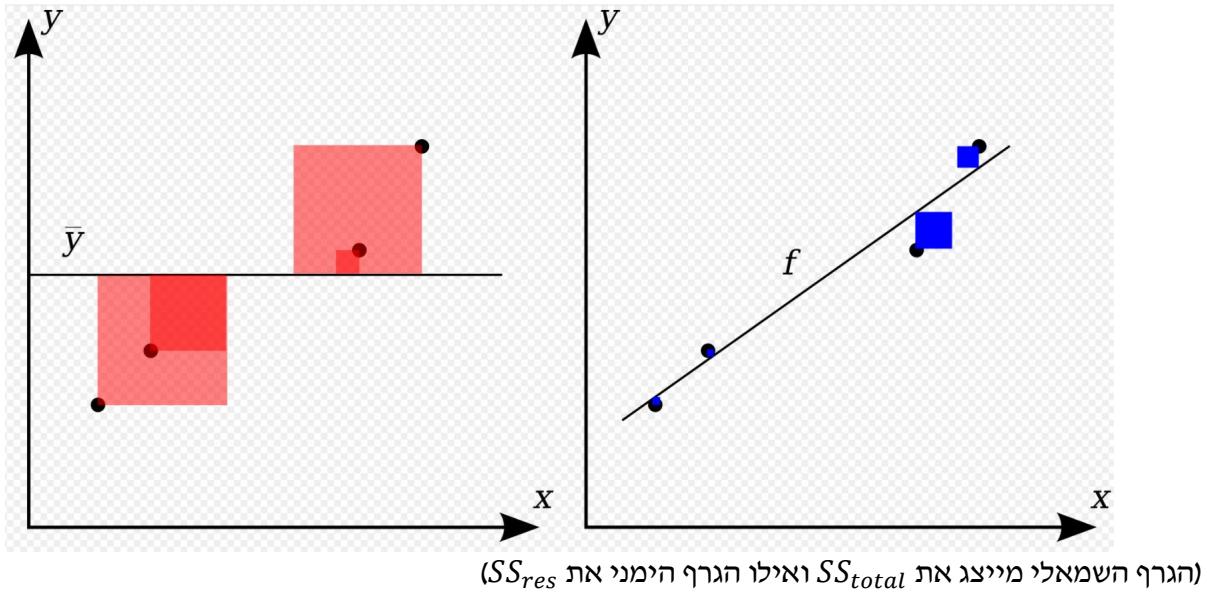
$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

תהי קבוצת ערכים $\{\hat{y}_n, \dots, \hat{y}_1\}$ המייצגים את ערכי החיזוי. נגדיר את המודל בו אנו עושים שימוש לצורך החיזוי להיות פונקציה $f(y_i) = \hat{y}_i$. נגדירשתי הגדרות חדשות:

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2$$

$$SS_{total} = \sum_i (y_i - \bar{y})^2$$

לצורך המחתת הגדרות אלה נציג שני גרפים המראים את החישובים אשר מבוצעות ו- SS_{res} ו- SS_{total} :



כעת, לצורך חישוב ה- $R - squared$ נשתמש בנוסחה :

$$R^2 = \frac{SS_{total} - SS_{res}}{SS_{total}} = 1 - \frac{SS_{res}}{SS_{total}}$$

נשים לב עתה לאבחנה הבאה. ככל שערך ה- SS_{res} יהיה קטן יותר, ככלمر הפרדיקציה לערכי $\{y_1 \dots y_n\}$ אותה העניק המודל תהיה קרובה יותר לעריכים האמיתיים של קבוצת מבחן זו, כך הביטוי $\frac{SS_{res}}{SS_{total}}$ ישאך ל-0 ו- R^2 ישאך ל-1. לכן נסיק מכך שהפונקציה *score* בה נעשה שימוש תחזיר עבורנו מספר גבוה יותר ככל שהמודל מעניק חיזוי טוב יותר. עבור אלגוריתמי החיזוי הקלאסיים (KNN, DT) מטריקה זו משמשת כברירת המחדל עבור ציון החיזוי.

מטריקה זו מוחשבת באופן הבא :

$$MedAE = median(|y_{true_1} - y_{pred_1}|, \dots, |y_{true_N} - y_{pred_N}|)$$

פונקציה זו דוגמת את החיצון מכלל השגיאות האבסולוטיות.

כפי שראינו לעיל, במצבים מסוימים יהיה לנו מאייד קשה להעריך את ביצועי המודלים באמצעות חלק מהמטריקות שהציגנו בשל אילוצים הקשורים לנוטונים אותם אנחנו עובדים. דוגמה בולטת לכך היא עבור אזורי בהם ערכי האמת y_{true} שוואפים לאפס. עבור ערכים אלו רוב המטריקות מתתקשות לתת ממד אמין ועלולות אף "להתפוץץ" (ערכים שוואפים לאינסוף). על מנת להעריך את המודלים שלנו גם במצבים אלו, ניתן להתבונן על תוחלת ערכי המבחן (mean test set mean) וממолов על תוחלת השגיאות (mean absolute error). התובנות על שני המדדים הללו מאפשר לנו לקבל אינדייקציה טובה לאיכות החיזוי שלנו. כמובן שנשאר שתוחלת השגיאות תהיה קטנה ככל הניתן, אך חשוב תמיד להתחשב גם בתוחלת ערכי המבחן שלנו. כך לדוגמה ערך $mae = 5$ לכשעצמו לא נותן לנו הרבה מידע, מכיוון שעבור מודל בו תוחלת ערכי המבחן

היא 1000 מדבר בשגיאה מאד נמוכה, אך עבור מודל בו תוחלת ערכי המבחן היא 10 מדבר בשגיאה משמעותית.

אלגוריתמי חיזוי קלאסיים

בחלק זה של הפרויקט השתמש באלגוריתמי חיזוי קלאסיים על מנת לפתור את בעיית החיזוי שלנו. האלגוריתמים בהם עוסק הם עצי החלטה (Decision Trees) ו-KNN. אלגוריתמים אלו משמשים לרוב לפתור בעיות סיווג, ואילו כאן ברכינו להשתמש בהם לפתור בעיה מסוג time series (בעיה שבה יש לנו תכונה שימושתנה כתלות בזמן). מטרתנו תהיה להבין מהן התאמות הנדרשות עבור כל אחד מהאלגוריתמים על מנת להתאיםם לפתור בעיית החיזוי שלנו, ולאחר מכן להציג שיפורים ואופטימיזציות לאלגוריתמים על מנת לשפר את דיוק החיזוי.

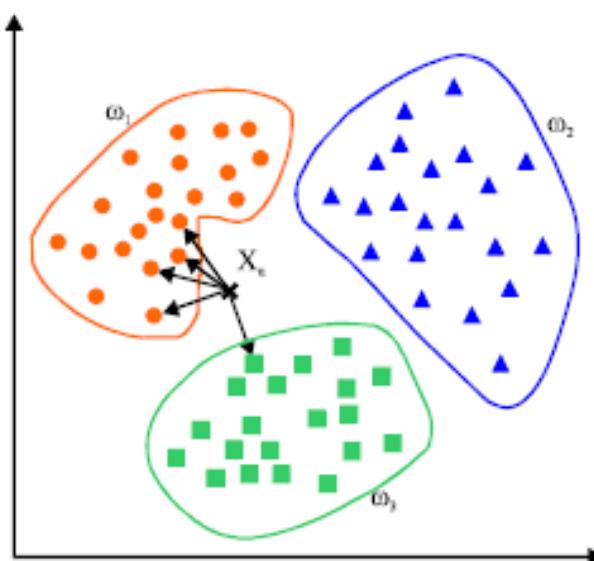
KNN

בפרק זה נמדל את הבעיה כבעיתת רגרסיה, אותה ניתן לפתור בעזרת ניטוח רגרסיה. בעיתת רגרסיה היא בעיה בה הערך אותו אנו ממחפשים הוא ערך רציף. ניתוח רגרסיה הוא שם כולל למשפחה של מודלים סטטיסטיים להערכת קשרים בין משתנים. למשתנה אותו אנחנו מנסים לנבא נקרא משתנה מושבר או משתנה תלוי. לתכונות שלנו נקרא המשתנים הבלתי תלויים או משתנים המנביאים. בחלק זה נראה פתרון לעוביה בעיתת KNN Regressor אלגוריתם KNN. תחילת נסביר על KNN.

נדיר את הבעיה שלנו (לצורך הסברת KNN) עבור בעיתת סיווג ולאחר מכן נסביר על KNN עבור בעיתת רגרסיה: נניח שיש לנו אוסף דוגמאות X , כאשר לכל $X \in X$ יש סיווג y . בזמן האימון אנחנו מקבלים אוסף של דוגמאות (y, x) , ובזמן הסיווג אנחנו מקבלים x עבורו אנחנו צריכים למצוא את y .

אלגוריתם K-Nearest Neighbors – KNN

ראשית נסביר את KNN עבור בעיתת סיווג, אז נסביר איך משתמש באלגוריתם KNN עבור בעיתת רגרסיה. אלגוריתם ה-KNN בזמן מייד מהחסן את כל הדוגמאות שהוא מקבל. כל דוגמא שלנו היא וקטור של תכונות.



עבור כל תכונה מספירתית $f_1(x), f_n(x)$, אנו משתמש בפונקציית מרחק אוקלידי $d_i(f_i(x), f_i(y)) = |f_i(x) - f_i(y)|$

$$d(x, y) = \sqrt{\sum_{i=1}^n d_i(f_i(x), f_i(y))^2}$$

ומרחק בין שני וקטורי תכונות הוא:

מכיוון שמרחק אוקלידי מושפע מהתחומים השונים של התכונות, נרצה לנормל את כל תחומי התכונות לטווח קבוע. בחרנו לנורמל לטווח 0-1.

בזמן סיוג, האלגוריתם מודד את המרחק בין האובייקט אותו יש לסוג לבין כל אחת מדוגמאות האימון (כאשר המרחק הוא סכום מרחקי התכונות) ומחזיר את הסיוג כתלות ב- K הדוגמאות הקרובות ביותר לאובייקט.

KNN הוא אלגוריתם למידה עצל (Lazy Learning) שזמן למידה לא מעבד את הדוגמאות, ורוב העבודה החישובית נעשית בזמן הסיוג.

אלגוריתם ה-KNN שומר את כל הדוגמאות בשלב הלמידה ואפשר להסתכל עליו כמרחב N ממדי (כאשר N הוא מספר הפיצרים) אשר דוגמאות הלמידה מפוזרות למרחב. כאשר מגיעה דוגמא לחיזוי, אנחנו שמים אותה למרחב ה- N ממדי, מסתכלים על K הדוגמאות הקרובות אליה ביותר, והערכץ עשה הדוגמא לקבל היא הסיוג של רוב K הדוגמאות הקרובות ביותר. ניתן גם למשקל את הדוגמאות, כך שambil K הדוגמאות הקרובות ביותר, נבדיל בין דוגמאות קרובות יותר לבין דוגמאות רחוקות יותר.

KNN Regression

זהו סוג של אלגוריתם KNN שמיועד לביעית הרגרסיבית שהציגנו בתחלת חלק זה.

אלגוריתם זה דומה מאוד ל-KNN כפי שהסבירנו זה עתה. ההבדל הוא שבביעיות רגרסיבית אין לנו מספר סופי של סיוגים עבור כל דוגמא באוסף שלנו, אלא לכל דוגמא יש ערך בטוח רציף. לכן, ב-k-NN, לאחר שבחרנו את k הדוגמאות הקרובות ביותר בזמן הסיוג, האלגוריתם מחשב את ערך ה- y לפי K הדוגמאות הקרובות לדוגמת הסיוג. הפונקציה הבסיסית של KNN Regression היא ממוצע K הדוגמאות הקרובות לדוגמת הסיוג, אך גם ניתן לבצע ממוצע משוקלל – דוגמאות קרובות יותר יקבלו משקל גבוה יותר – בממוצע בחישוב y .

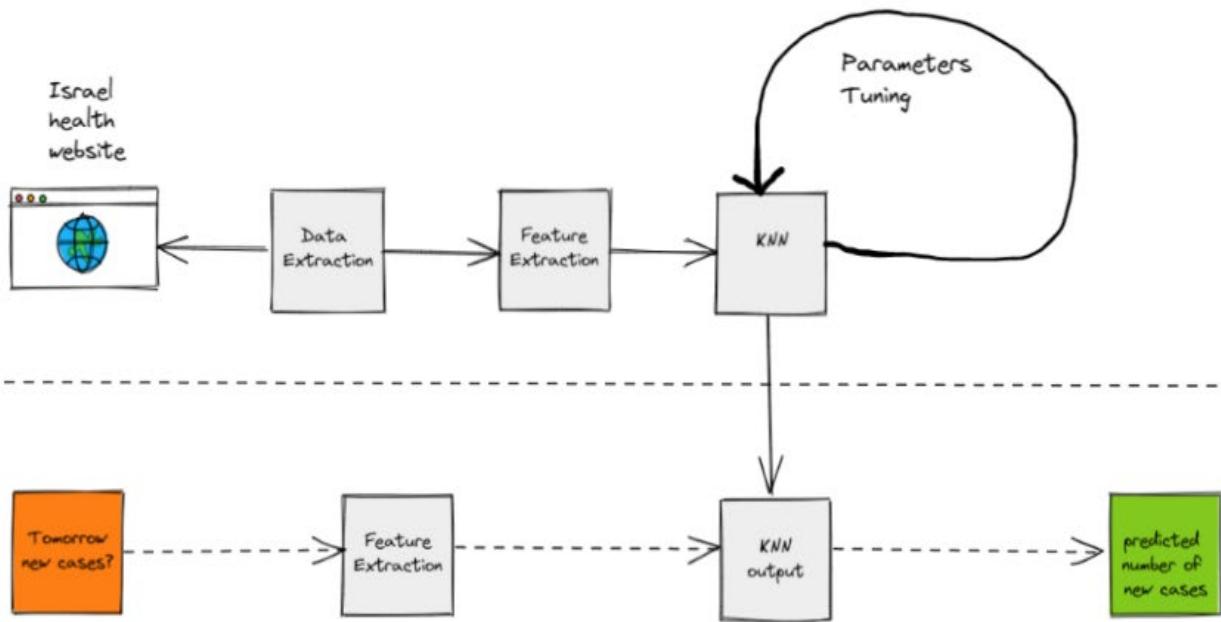
כעת נסביר איך השתמש ב-KNN עבור הבעה שלנו, וממה מרכיבים x ו- y שלנו:

כל דוגמא – x מרכיבת מ- N תכונות (כל תכונות שהסבירו בפרק ה-Data). עבור כל דוגמא יש לנו את ערך ה- y – שם העמודה שלו היה today_verified_cases.todayVerifiedCases. זהו הערך הרציף אותו אנחנו מנסים לחזות – מספר החוליםים היומי.

מדוע לדעתנו KNN טוב לפתרון הבעיה?

בחרנו באלגוריתם KNN לפתרון הבעיה מושם שהוא אלגוריתם פשוט, וכי שנטקלנו רבות בתחום מדעי המחשב יש יתרון גדול לפתרון פשוט. אלגוריתם זה בעצם מביא את ערך התצפית בהתבסס על התכונות, ככלומר הוא מבטא את הקשר בין התכונות לבין ערך התצפית. אנחנו משערים שבבעיה חיזוי הקורונה, מספר החוליםים היומי קשור מאוד לתכונות אותן בנוינו, ולכן אנחנו חושבים שאלגוריתם זה ייבן לנו תוצאות טובות.

תרשים המערכת :



הסבר המערכת :

- **Data Extraction** – בשלב זה הורדנו מה API של משרד הבריאות את הטבלאות והנתונים של הקורונה בישראל. הסבר מפורט על הטבלאות נמצא בפרק הצגת המקורות.
- **Feature Extraction** – בשלב זה ביצענו מניפולציות על ה data כדי להמיר אותו למבנה טבלאי שמתאים לאלגוריתם KNN. ה Data בטבלאות היה מסודר כך שככל רשותה (יישוב + תאריך) יש את מספר החוליםים/מחוסנים המצביע מתחילת הקורונה. בניית זה לא טוב לאלגוריתם שלנו מכיוון שאנחנו רצינו להסתכל על מספר החוליםים ומחוסנים יומי ובעזרתם לחזות את מספר החוליםים היומי. לכן בשלב זה חילצנו מהנתונים את מספרי החוליםים היומיים, המוחסנים היומיים וכדומה. (כמפורט בפרק ה-Data) בשלב זה בנוסח נרמלנו את ה data לפי גודל היישוב על מנת שהנתונים של ערים בגודלים שונים יעוזו אחת לשנייה.
- **KNN** – נשתמש באלגוריתם KNN המוכן מתוך ספירת הפיתון של Scikit-learn ונאמן אליו בעזרת ה data set שבנוינו בשלבים הקודמים.
- **KNN Output** – זהו התוצר משלב הלמידה (החלק העליון של האיור). KNN Output הוא אלגוריתם KNN מאומן (מכיל אוסף דוגמאות אימון) אשר הפרמטרים שלו מכונים.

מתודולוגיה ניסויית

בשלב זה נרצה לאמן מספר אלמנטים שונים באלגוריתם :

- Feature selection – כרגע בבעיה יש לנו 33 תכונות. כפי שלמדנו בקורס מבוא לבינה מלאכותית מספר רב כל כך של ממדים עלול לפגוע ביציעים שלנו (Curse of dimensionality). KNN הוא אלגוריתם שרגיש לעביה זו מכיוון שריבוי התכונות משਬש את המרחק האוקלידי בין הדוגמאות השונות במרחב. ניתן לדמיין זאת, אם נחשוב על שני וקטורים שנרצה שיהיו קרובים אחד לשני וניתן לזהות זאת על ידי מספר קטן של תכונות. נניח שנסתכל על סט תכונות המכיל את התכונות החשובות, אך מכיל תכונות רבות נוספת. אז הווקטורים יראו לנו במרחב אחד מהשני בגל השוני בתכונות הלא חשובות. אם נכליל את רעיון זה נוכל לדמיין שאם נסתכל על מרחב תכונות מאוד גדול, כאשר אנחנו מודדים מרחק אוקלידי בסכום המרחקים האוקלידיים של התכונות כפי שהסביר לעיל, ייתכן כי כל הדוגמאות יראו במרחב דומה אחת לשניה. לכן נרצה להוריד תכונות מיותרות, ובכך המרחק האוקלידי שלנו יהיה בין התכונות החשובות.
- אנחנו בחרנו לבצע בחירת תכונות על ידי בחירת M תכונות רנדומליות (כאשר M הוא פרמטר שאנו מאמנים), מדידת דיקט האלגוריתם, ולאחר ניסויים רבים בחירת M התכונות שהניבו את התוצאה הטובה ביותר.
- אימון ההיפר פרמטר K (פרמטר של האלגוריתם). בחירה זו היא כמובן חשובה מאוד באלגוריתם K-NN. ערכים גבוהים של K גורמים לצמצום ההשפעה של רעש על פلت ה KNN. מצד שני, אם ממדים נאים את המרחק אז ערך K גבוה גורם לחיקת המרחב להיות פחות מובהקת. נציין שגם ממדים נאים בבעיות רגסיה ולא בעיות סיוג או ערך K לא צריך להיות אי זוגי כפי שנלמד בקורס, מכיוון שהאלגוריתם נותן פلت משוקלל של K השכנים של דוגמת המבחן.
- פונקציית המשקל – פונקציה זו מושפעת על המשקל של ערך ה- y שיבחר בזמן ה-Predict, כאשר היא בוחרת כיצד למשקל את K הדוגמאות הקרובות לדוגמת הבוחן.
 - Uniform – אשר נותנת לכל אחת מ K הדוגמאות משקל זהה.
 - Distance – אשר נותנת לכל דוגמא משקל לפי המרחק שלה מדוגמת החיזוי.
- טרנספורמציה לבעיית rolling average – מטרתנו בחלק זה היא "לנקות" את מספר החוליםיים היומיי כפונקציה של הזמן מ-"רעים". לדוגמא, בכל ימי ישנו שבת יש פחות בדיקות ולבן הפונקציה לא חלקה. קשה לחזות קפיצות כלשה בפונקציה, וכמובן שמספר בדיקות נמוך ביום שבת לא מעיד על מספר החוליםים נמוך, אלא על מספר בדיקות נמוך.

אימון האלגוריתם

מכיוון שיש לנו הרבה דברים אותם נרצה לאמן, משתמש במספר שיטות :

1. משתמש במתודולוגיה הניטויים שלנו במאפיינים רנדומליים.
2. נרץ ניסויים על כל אימון בנפרד, מכיוון שמכפלה קרוטזית שלהם כדי למצוא את הקומבינציה הטובה ביותר תהיה מאוד יקרה חישובית. לעומת נאמן על K בנפרד, נאמן על feature selection בנפרד וצדומה.

אופן האימון :

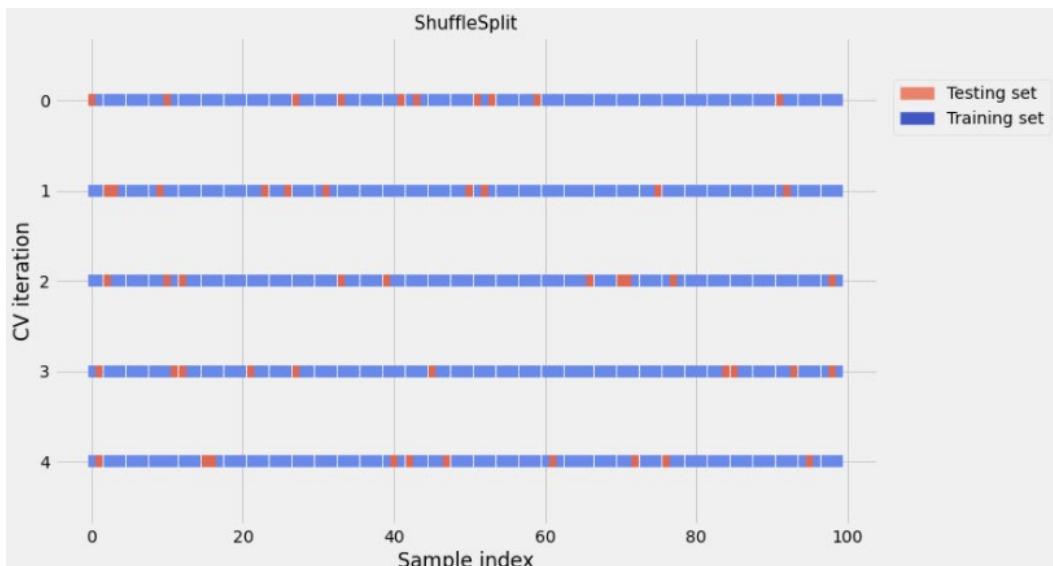
הינו רוצים לבצע אימון על כל אחת מהנקודות שהצגנו לעיל ולהשתמש ב-Cross Validation כדי לקבל דיוק אמין יותר מהריצה בודדת על train ו-test. כאשר ניסינו להשתמש ב-Cross Validation נתקלו במספר בעיות שנובעות מכך שהבעה שלנו היא בעיית time series, אשר Cross Validation פחות מתאימה לה (בשימושים הסטנדרטיים).

גם כאשר השתמשנו ב-Cross Validation אשר מיועד ל-time series, נתקלו בתוצאות פחות טובות, מכיוון שתוחלת ערך ה- \bar{y} של test-set משפיע בצורה ניכרת על האלגוריתם. עצת נסביר על שיטת ה-Cross Validation השונות שניסינו ומדובר בכך לא מתאימות לעבודה בעיה שלנו.

בשלב ראשון לקחנו את dataset שלנו ופיצלנו אותו ל-train ו-test. בשלב הבא רצינו להשתמש ב-K Cross Validation כדי לקבל דיוק אמין יותר שלא תלוי ב train ו-test ספציפיים. תחילת, השתמשנו ב-K Cross Validation רגיל, שככל פעם בחר קבוצה אחת להיות ה-test ושאר הקבוצות היו ה-train. לאחר שקיבלו תוצאות דיוק מרשים מאוד (של עד 98%), הבנו מיד שיש בעיה מהותית. לאחר מחקר מעמיק מצאנו שהבעה שלנו נובעת מהניסיון לפטור בעיית time series עם שיטת K Cross Validation כללית שלא מיועדת לפתור בעיות מסווג זה.

.K cross validation+ Shuffle Split

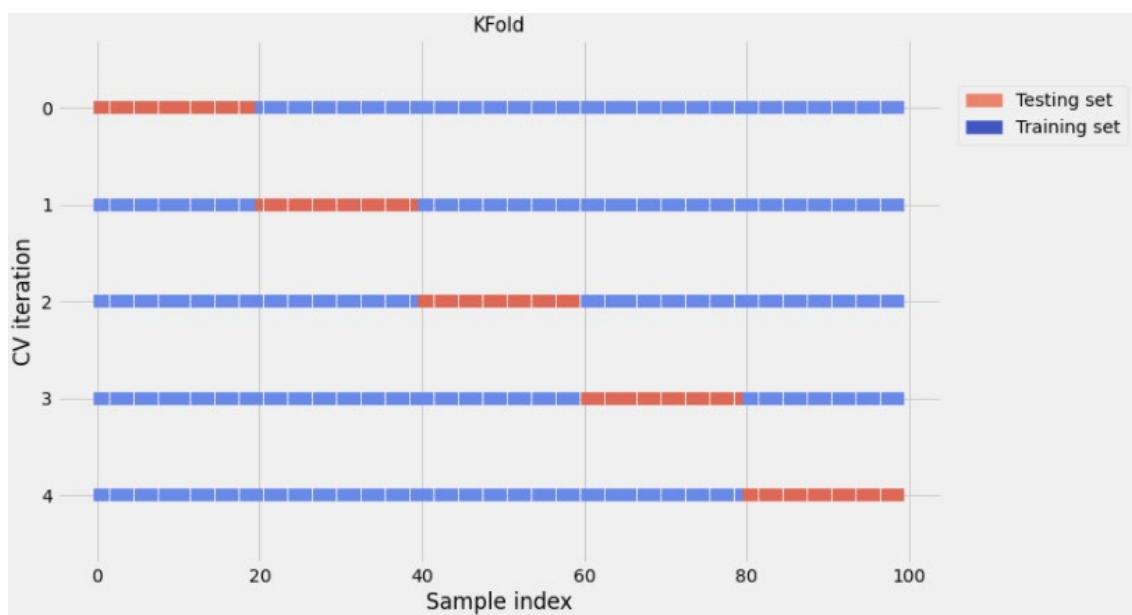
גרסה זו בוחרת בכל איטרציה את test-set מתוך ה-train-set כך שה-test שזור למחרי בתוך ה-train. לעומת ה- Shuffle Split : K cross validation+ Shuffle Split



הסבר הגרף : בציר האופקי אנחנו רואים את מספרי הדוגמאות ב-`train` (שמסודרות אצלנו בסדר כרונולוגי על ציר הזמן), ובציר האנכי אנחנו רואים 5 איטרציות שונות, אשר בכל אחת מהן `train`-`test` שונה. נציג את הניתוח שלנו וההסבר מדוע קיבלנו דיקוק גבוה מאוד בשימוש בשיטה זו. עבור כל דוגמת מבחן, K השכנים הקרובים שלה היו הדוגמאות מימינה ומשמאליה וכן המוצע שלהם נתן עד כדי שגיאת זניחה את מספר החוליםים המדויק. לדוגמה אם אנחנו יודעים שביום 1- t מסויים היו 2 חולים, וביום $t+1$ יהיו 6 חולים, אז סטוטיסטיות ביום t יהיו 4 חולים. עבור $k=6$ בהנחה שנבחרו k דוגמאות כך ש $\frac{1}{k}$ יהיה לפני הדוגמא ו- $\frac{2}{k}$ יהיה אחרי הדוגמא, אז השגיאה אפיו תקטן עוד ולכון `K cross validation + Shuffle Split` לא מתאים עבור אימון KNN לביעית Time Series.

לאחר מכן, ניסינו להשתמש בגרסת `KFold` קלאסי. בגרסה זו בכל איטרציה ה-`test set` רציף, כלומר כל דוגמאות המבחן צמודות אחת לשניה.

: `K cross validation - KFold`

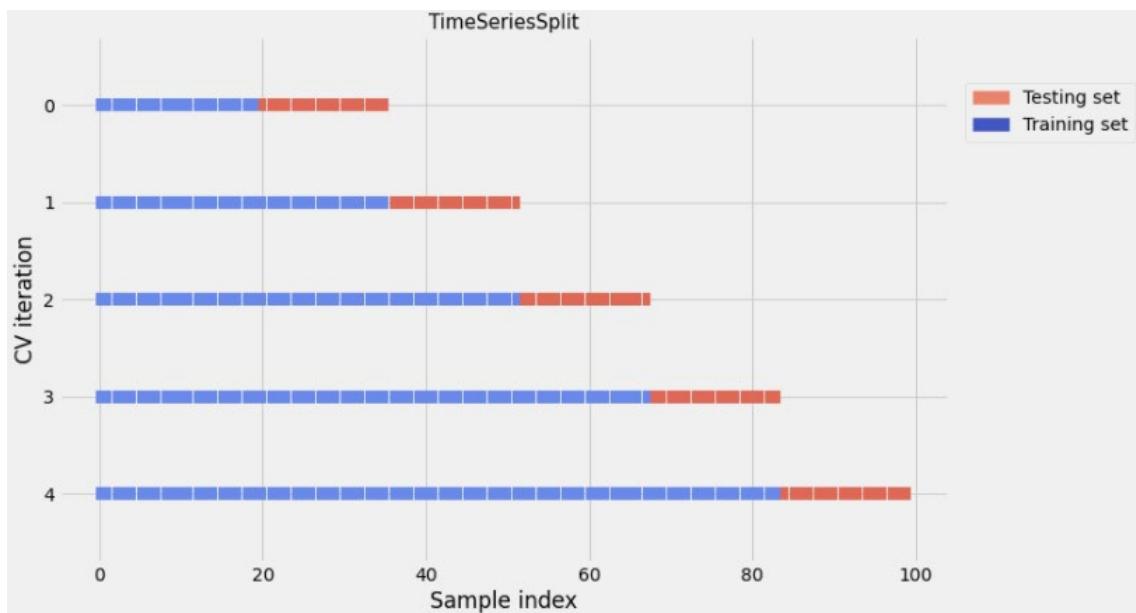


הסבר הגרף : בדומה לגרף הקודם, בציר האופקי יש לנו את מספרי הדוגמאות (שמסודרות אצלנו בסדר כרונולוגי על ציר הזמן) ובציר האנכי יש לנו 4 איטרציות כאשר בכל אחת ה-`test set` שונה.

לאחר הרצת גרסה זו וניתוח של התוצאות הבנו שגם יש בעיה. ניתן לראות שבאייטרציות 3-0 אנחנו מבצעים חיזוי לימים מסוימים בעזרת הימים הבאים, כלומר בעזרת העתיד. כמובן שהזה מצב לא תקין ולכן גם ה-`KFold` לא ניתן להשתמש.

לאחר מכן, ניסינו לבצע `K Cross Validation` בשיטת `TimeSeriesSplit`. בשיטה זו, באיטרציה הראשונה נבחר בлок ראשון של דוגמאות `train` ומיד לאחר מכן בлок דוגמאות `test`. באיטרציה הבאה ניקח 2 בלוקי `train` ובלוק אחד `test` וכך הלאה.

תיאור גרפי ל-K cross validation - TimeSeriesSplit



הסבר הגרף: בדוגמה לgraf הקודם, בציר האופקי יש לנו את מספרי הדוגמאות (শমসূদ্রোত אצלנו בסדר הכרונולוגי על ציר הזמן) ובציר האנכי יש לנו 4 איטרציות כאשר בכל אחת ה-test set הוא בлок שצמוד לבlok train הולך וגדל.

בשיטת זו כל הביעיות הקודמות שלנו נפתרו, וכפי שניתן לראות אנו לא משתמשים בדוגמאות מהעתיד כדי לחזות את ההווה, אך בשיטה זו נתקלנו בבעיה חדשה. דיקוק האלגוריתם שלנו רגש מאוד לתוחלת ה-test set (של ערכי ה- y).

בתקופות כאשר מספרי חולי הקורונה נמוכים, אז תוחלת ערכי ה- y שלנו קרובה מאוד ל-0. ניתן לראות שכאשר אנו משתמשים ב-K Cross Validation Time Series Split, אז עברו כל בлок אשר ה- y שלו שווה ל-0, אנחנו מקבלים דיקוק רע מאוד.

נציג פלט לדוגמא של הרצת K Cross Validation Time Series Split כאשר $K=10$ (כלומר יש 10 ניסויים כאשר ה-test set זו כפיה שתואר לעיל).

בכל שורה נציג את דיקוק R₂, את השגיאה MAE, ואת תוחלת ערכי y test set

- knn accuracy: R₂: 0.764, mae: 10.509, y_test_mean=18.568
- knn accuracy: R₂: -1.422, mae: 11.354, y_test_mean=10.807
- knn accuracy: R₂: -20.071, mae: 10.591, y_test_mean=2.087
- knn accuracy: R₂: -133.89, mae: 4.58, y_test_mean=0.4
- knn accuracy: R₂: -26.618, mae: 0.756, y_test_mean=0.105
- knn accuracy: R₂: -2.139, mae: 0.236, y_test_mean=0.058
- knn accuracy: R₂: -7.813, mae: 1.573, y_test_mean=1.012
- knn accuracy: R₂: 0.428, mae: 3.767, y_test_mean=5.873

- knn accuracy: R2: 0.564, mae: 11.772, y_test_mean=22.761
- knn accuracy: R2: 0.86, mae: 8.794, y_test_mean=26.453
- ❖ avg accuracy: -18.933645416613896

ניתן לראות שברירות כאשר y_{test_mean} קרוב ל-0, אנחנו מקבלים דיק R2 שלילי גבוה מאוד. זאת מכיוון שכפי שהראינו בתיאור מטריות המדידה ב-R2 כדי לא לחלק ב-0, אנחנו מוסיפים למיננו את y_{test_mean} , וכך הוא קרוב ל-0, אנחנו מקבלים מספרים שליליים גדולים במערכות המוחלט. לכן, הממוצע של ה-Cross Validation שקיבלו לא שימושי.

Feature Selection

בשלב הבא ביצענו Feature Selection, נתאר את הניסוי אותו ביצענו:

- עבור כל M עד 15 :
 - בחר M תכונות רנדומליות.
 - מדוד את ביצועיו של KNN לפי $R2$ score, עם M התכונות הנ"ל.
- בחר את קבוצת התכונות שננתנה את הדיק הטוב ביותר.

לאחר מספר פעמים שהרצנו את הניסוי הנ"ל קיבלו כי התכונות הטובות ביותר ביוטר שקיבלו הן:

vaccinated_dose_3_total, dose_3_in_last_2_week, verified_cases_7_days_ago, City_Code,
verified_cases_14_days_ago, colour

עבורן קיבלו את הדיק הבא:

- ❖ knn accuracy: R2: 0.802, mae: 7.664, $y_{test_mean}=21.038$

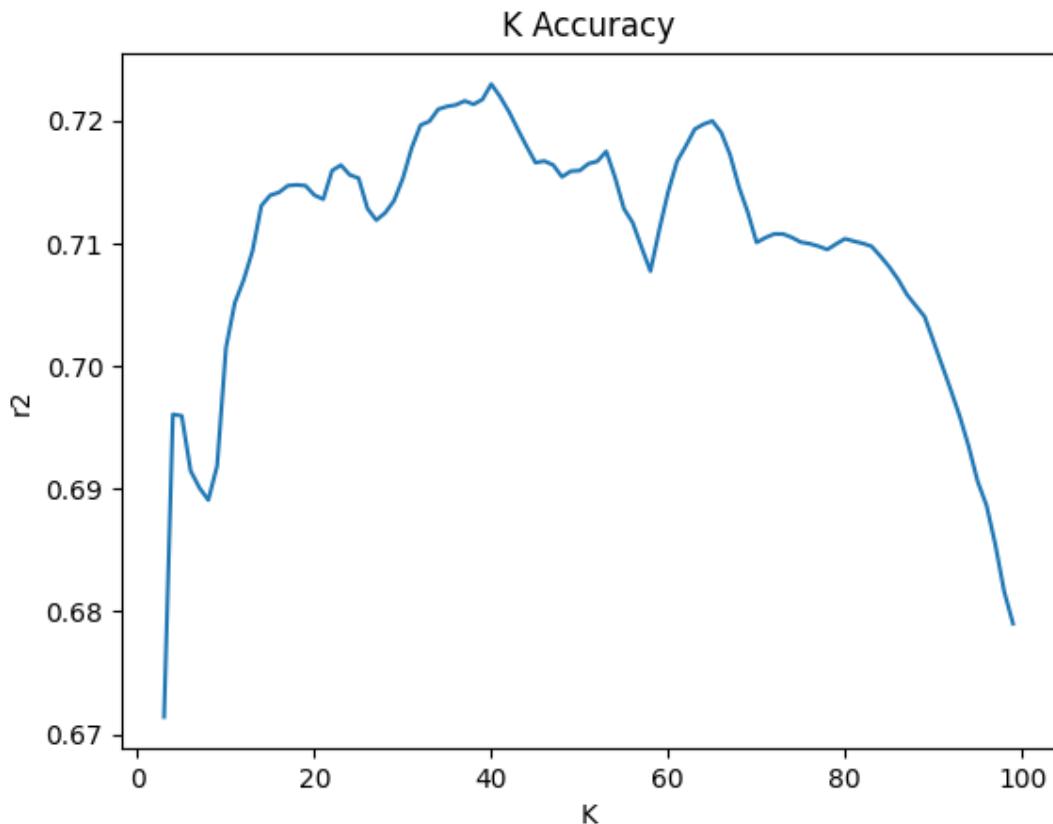
ניתן לראות שזה שיפור ביצועים משמעותית מהדיק לפני ה-Feature selection שהוא:

- ❖ knn accuracy: R2: 0.658, mae: 10.988, $y_{test_mean}=21.038$

בנוסף נציגו שעבור 33 תכונות, זמן הריצה היה ארוך משמעותית (פי כמה) מריצה עם מספר נמוך של תכונות. זאת מכיוון שעבור כל דוגמת מבן אנחנו מבצעים השוואה לעשרות אלפי דוגמאות אימון. לכן כאשר מורידים את מספר התכונות, אז השוואת שני וקטורי תכונות מהיר משמעותית.

אימון K

ביצעו ניסוי על פרמטר ה-K של האלגוריתם. קיבלנו שעבור $K=40$ אנחנו מקבלים את הדיק הגבוה ביותר.



בחירה פונקציית משקל

ביצעו ניסוי שבודק את פונקציות המשקל. ראשית השתמשו בפונקציה הדיפולית של האלגוריתם – uniform. פונקציה זו מושקלת באופן שווה את K הדוגמאות הקרובות לדוגמת המבחן כאשר היא מחשבת את הפלט של דוגמת המבחן.

לאחר מכן, בדקנו את פונקציית distance -distance שמשקלת את מרחק הדוגמת לימון מדוגמת המבחן. דוגמאות שקרובות לדוגמת המבחן יקבלו משקל גבוה בחישוב התוצאה לדוגמת המבחן, ודוגמאות רחוקות יקבלו משקל נמוך יותר (כל אחת מ K הדוגמאות מקבלת משקל שונה בחישוב תוצאה דוגמת המבחן).

קיבלו דיק גבוה יותר עם פונקציית משקל ה-distance.

testing best weights

uniform:

❖ knn accuracy: R2: 0.723, mae: 12.139, y_test_mean=24.934

distance:

- ❖ knn accuracy: R2: 0.742, mae: 11.732, y_test_mean=24.934

תוצאות וסיכום

לפני האימון וכל הניסויים, עבור ריצת KNN עם כל העמודות עם k רנדומלי שבחרנו (k=10) קיבלנו score של 0.587

לאחר כל שלבי האימון קיבלנו דיווק גבוה מאוד באופן ניכר של R2=0.742!

פלט השוואת התוצאות:

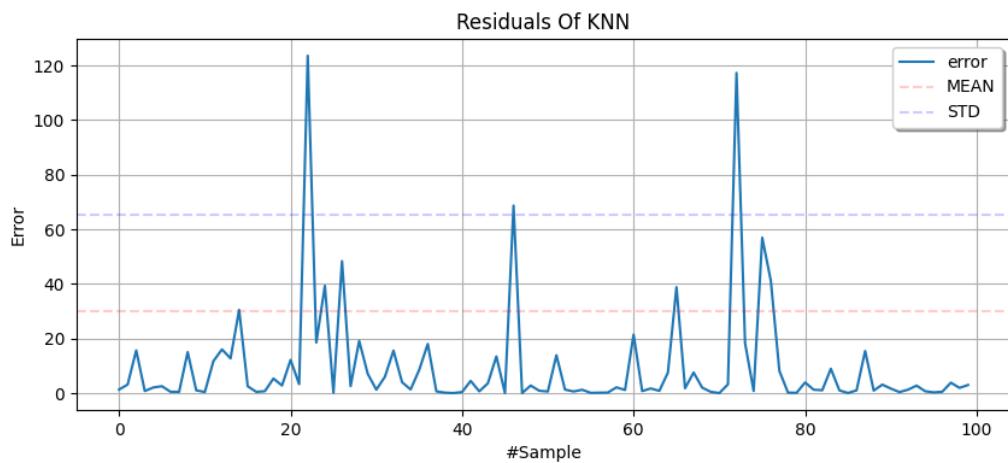
Train date range: 05-01-2021 - 31-08-2021

Test date range: 01-09-2021 - 25-09-2021

test mean: 24.934

- ❖ knn all columns: R2: 0.587, mae: 15.325, y_test_mean=24.934, test_len=7025
- ❖ knn after learning: R2: 0.742, mae: 11.732, y_test_mean=24.934, test_len=7025

נציג את גראף ה-residuals (גרף המציג את ההפרשיות בין ערך האמת לערך החיזוי)



ניתן לראות שרוב הדוגמאות השגיאה שלנו קטנה ושבמעט מאוד דוגמאות השגיאה מאוד גדולה.
השיפור הגדול ביוטר בביצועי האלגוריתם התקבל מבחירה התכונות.

בשלב זה כדי להסביר את התוצאות השתמשנו במודל המאומן, כדי לנתח את תוצאותיו ולהבין אם אנחנו טובים באופן מיוחד בתחום ספציפי, או לחילופין לא טובים במקרה ספציפי.
הרצינו מספר ניסויים, כאשר בכל ניסוי בחרנו את ה-test שלנו להיות בעל מאפיין ספציפי על מנת להבין איך אנחנו טובים יותר ואייה טובים פחות.

המאפיינים אותם ניסינו לבדוק הם:

- ישובים גדולים.
- ישובים קטנים.
- ישובים עם מספר חולים גבוה.
- מספר חולים נמוך.
- ערים ירוקות/צחובות/כתומות/אדומות.

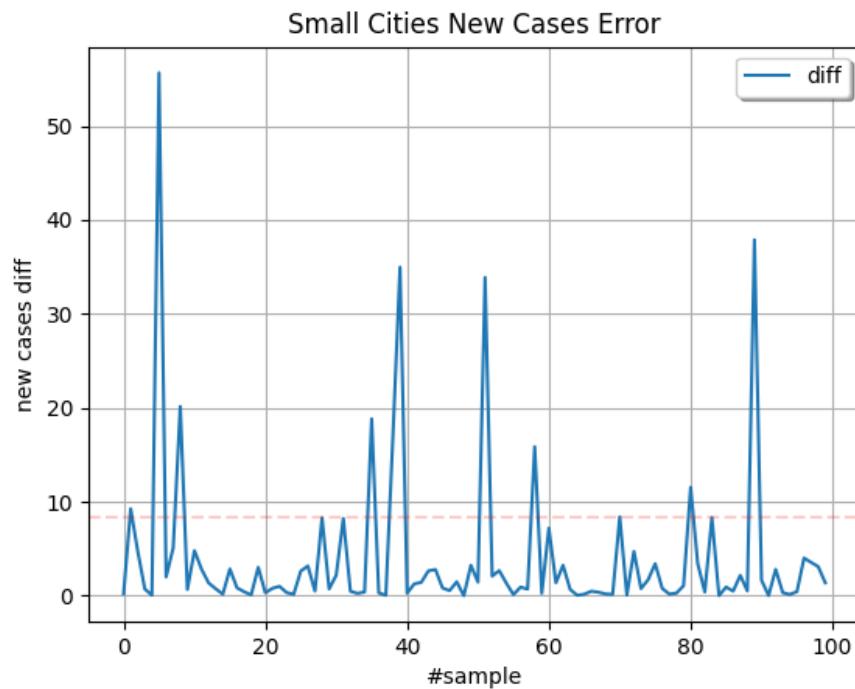
כפי שהסבירנו, בכל ניסוי כזה (לכל ניסוי יש שורת פלט מתחת להסביר זה) לקחנו את ה-test set שלנו ובחרנו subset לפי התכוונה המסוימת. לדוגמה עבור ישובים גדולים, לקחנו ישובים עם יותר מ-30,000-35,000 אנשים. ערכי ההפרדה בין ישובים גדולים לקטנים, ומספר חולים נמוך וגובה נקבעו לפי חקירת ה-data.

- knn_on_small_cities population < 30000: R2: 0.282, mae: 5.088, y_test_mean=8.376
- knn_on_big_cities population > 30000: R2: 0.618, mae: 37.045, y_test_mean=83.422
- knn_on_small_new_cases new_cases < 50: R2: -0.3, mae: 6.261, y_test_mean=10.052
- knn_on_big_new_cases new_cases > 50: R2: 0.518, mae: 53.294, y_test_mean=129.31
- knn_on_colour colour=green: R2: 0.316, mae: 0.877, y_test_mean=1.05
- knn_on_colour colour=yellow: R2: 0.829, mae: 5.64, y_test_mean=13.002
- knn_on_colour colour=orange: R2: 0.769, mae: 14.257, y_test_mean=34.057
- knn_on_colour colour=red: R2: 0.57, mae: 16.867, y_test_mean=29.308

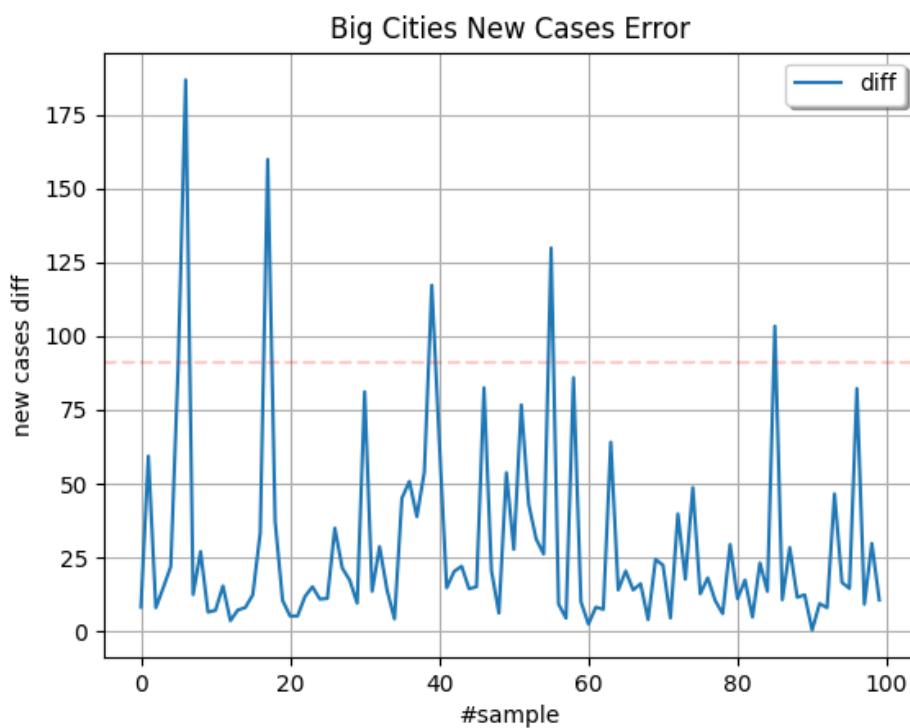
ניתן לראות שדיוק R2 מאוד מושפע מתחולת המספרים. כאשר y_test_mean נמוך יותר R2 Score נמוך יותר (כפי שכבר הסבירנו).

ניתן לראות באופן ברור שבישובים קטנים (בهم יש מספרי חולים נמוכים יותר), בימים עם מספרי חולים קטנים, ובערים ירוקות דיוק-h2 score נמוך משמעותית. לעומת זאת, ניתן לראות באמצעות MAE השגיאה שלנו לא מאד גדול. לדוגמה, בישובים קטנים יש 8 חולים בממוצע. כאשר אנחנו טועים ב-5 חולים בממוצע, זו טעות שאומנם סטטיסטית היא גדולה, אבל בעולם האמיתי (בביעית חיזוי מספרי הקורונה) היא זניחה. ולכן למרות ש- R2 Score שלנו נמוך, אנחנו מרווחים מההוואות שקיבלנו.

נראה לדוגמה את גרפ' השגיאה (בערך ממוצע) של ניסוי על הערים הקטנות. זהה השגיאה של 100 דוגמאות מבחן. ניתן לראות שרובן קטנות, וקטנות מהתוחלת של \bar{Y} (הקו אדום) :



גם עבור ערים גדולות, ניתן לראות שהשגיאות שלנו נמוכות יחסית לתוחלת :



Decision Tree

רקע תאורטי

בפרק זה נעשה שימוש במודל המכונה "עץ החלטה". פרק זה מניח ידע מקדים במושגים כלליים הנוגעים לפתרות בעיית חיזוי באמצעות עצי החלטה. בפרויקט שלנו אנו עושים שימוש ברגרסור להבדיל ממושג אותו למדנו בקורס "מבוא לבינה מלאכותית". נציג קצת מהו רגרסור.

באשר לעצי החלטה קיימים שני מודלים עיקריים :

- עץ סיווג – עץ אשר בהינתן דוגמא, מחזיר סיווג בינארי (*True/False*) או נומינלי.
- עץ רgressיה – עץ אשר בהינתן דוגמא, מחזיר ערך מסווג מטוחה רציף. מודל זה מכונה רגרסור שבו נעשה שימוש לכל אורך הפרק הנוכחי.

נפרט קצת על הרגרסור שזה עתה הצגנו.

במקרה שלנו, בהינתן יישוב אשר נרצה לחזות עבורו את מספר חוליות הקורונה ביום מסוים, רגרסור זה יחזיר תוצאה אשר תיתן הערכה מסוימת למספר חוליות הקורונה ביום המבוקש. כידוע, בעץ החלטה קיים שורש העץ וממנו מבוצעים פיצולים לצמתים עד להגעה לעלה שבו למעשה נמצא הסיווג. להבדיל מעץ ההחלטה אשר נותן סיווג בינארי (*True/False*) או נומינלי, בעץ ההחלטה בו אנו עושים שימוש בפרק זה מניב כאמור תוצאה מסוימת רציף. נספק קצת הסבר על תהליך בניית העץ או במילים אחרות, נסביר כיצד בהינתן צומת אב, מבוצעת החלוקה לשני צמותי בן. כאשר אנו מתבוננים בצומת בעץ, צומת זה מכיל למעשה דוגמאות אשר לכל אחת מהן סיווג מסוים. כאשר האלגוריתם מנסה לפצל את צומת האב לשני צמותי בן, נעשה שימוש ב- Variance Reduction. עבור כל תכונה, מתבצעת חלוקה של דוגמאות האב לשני צמותי בן. עבור כל חלוקה שכוו מתבצע החישוב הבא :

$$Var(node) = \frac{1}{n} \sum_{i=0}^n (y_i - \bar{y})^2$$

$n = \text{Number of samples}$, $y_i = i^{\text{th}} \text{ value}$, $\bar{y} = \text{Node } y_i \text{'s Values Average}$.

$$Var(Reduction) = Var(parent) - \sum w_i Var(child_i)$$

$$w_i = \frac{\#Child_{Samples}}{\#Parent_{Samples}}.$$

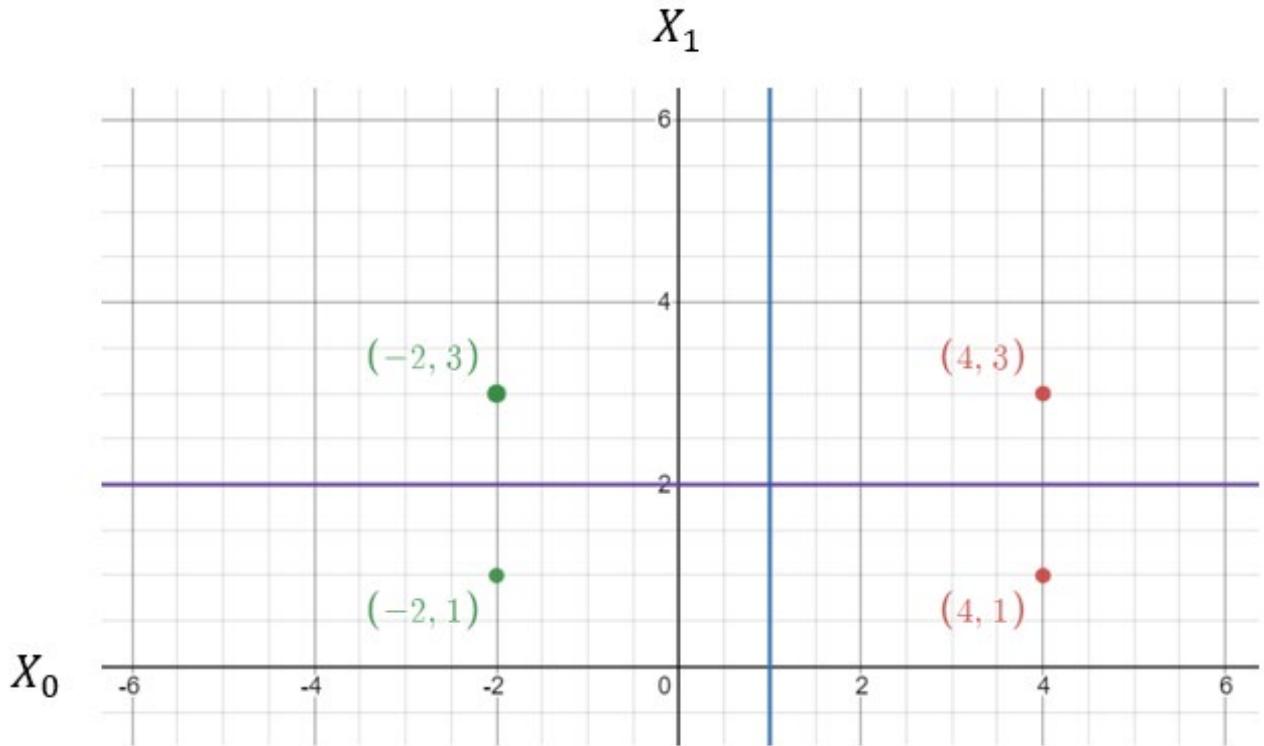
האלגוריתם יבחר תכונה לפחות אחת שתאפשר את התוצאה הגבוהה ביותר עבור החישוב הניל. נמchioש את האמור באמצעות דוגמא :

נניח שקיימות לנו 4 דוגמאות מהן נרצה לבנות את העץ. לכל דוגמא קיימות שתי תכונות שננסמן x_0 , x_1 וסיווג מסומי שננסמו ב-ע. בגרף המוצג מטה, צבעי הדוגמאות יוגדרו באופן הבא :

$$Y_{Red} = 10, Y_{Green} = 1$$

כאשר הצבע מסמל למעשה את הערך ע.

להלן הגרף:



כעת, נבצע שתי חלוקות שוונות של צומת האב לשני צמתים בן, כל פעם באמצעות תכונה באמצעות חלוקה דינמית באופן הבא:

$$parent = \{(-2,3), (-2,1), (4,1), (4,3)\}$$

$$x_0 \leq 1 \rightarrow left_{node} = \{(-2,3), (-2,1)\}, right_{node} = \{(4,1), (4,3)\}$$

$$x_1 \leq 2 \rightarrow left_{node} = \{(-2,3), (4,3)\}, right_{node} = \{(-2,1), (4,1)\}$$

בاهינתן שתי חלוקות הניל', נרצה להחליט איזו חלוקה תניב עבורנו פיצול טוב יותר. נחשב את השוונויות לכל אחד מהצמתים:

$$\begin{aligned} Var(parent) &= \frac{1}{4} \sum_{i=0}^4 (y_i - \bar{y})^2 \left(y_1, y_2 = 1, y_3, y_4 = 10, \bar{y} = \frac{1+1+10+10}{4} = 5.5 \right) \\ &= \frac{1}{4} * [(1-5.5)^2 + (1-5.5)^2 + (10-5.5)^2 + (10-5.5)^2 = 81 \end{aligned}$$

$$x_0 \leq 1 \rightarrow Var(left_{node}) = \frac{1}{2} \sum_{i=0}^2 (y_i - \bar{y})^2 \left(y_1 = 1, y_2 = 1, \bar{y} = \frac{1+1}{2} = 1 \right)$$

$$= \frac{1}{2} * [(1-1)^2 + (1-1)^2] = 0$$

$$x_0 \leq 1 \rightarrow Var(right_{node}) = \frac{1}{2} \sum_{i=0}^2 (y_i - \bar{y})^2 \left(y_1 = 10, y_2 = 10, \bar{y} = \frac{10+10}{2} = 10 \right)$$

$$= \frac{1}{2} * [(10-10)^2 + (10-10)^2] = 0$$

$$x_1 \leq 2 \rightarrow Var(left_{node}) = \frac{1}{2} \sum_{i=0}^2 (y_i - \bar{y})^2 \left(y_1 = 1, y_2 = 10, \bar{y} = \frac{1+10}{2} = 5.5 \right)$$

$$= \frac{1}{2} * [(1-5.5)^2 + (10-5.5)^2] = 81$$

$$x_1 \leq 2 \rightarrow Var(right_{node}) = \frac{1}{2} \sum_{i=0}^2 (y_i - \bar{y})^2 \left(y_1 = 1, y_2 = 10, \bar{y} = \frac{1+10}{2} = 5.5 \right)$$

$$= \frac{1}{2} * [(1-5.5)^2 + (10-5.5)^2] = 81$$

כעת נבעצע את החישוב הנזכר לעיל. נקבל:

$$Var_{x_0}(Reduction) = Var(parent) - \sum w_i Var(child_i) = 81 - \frac{2}{4} * 0 - \frac{2}{4} * 0 = 81$$

$$Var_{x_1}(Reduction) = Var(parent) - \sum w_i Var(child_i) = 81 - \frac{2}{4} * 81 - \frac{2}{4} * 81 = 0$$

ניתן לראות שהחלוקת הראשונה מניבה עבורנו תוצאה גבוהה יותר מאשר חלוקה השנייה ולכן נבחר בחלוקת זו. ניתן לראות בנוסף שאנוחלוקת הראשונה מייצרת שני צמתים בין אשר השמאלי שביניהם מכיל דוגמאות בעלות צבע יירוק בלבד כולם סיוג בעל ערך 1 ואילו הבן הימני מכיל דוגמאות בעלות צבע אדום בלבד המייצג סיוג בעל ערך השווה ל-10. לעומת זאת,חלוקת השנייה מייצרת שני צמתים בין אשר בכל אחד מהם קיימת דוגמא ירוקה ואדומה מה שהשאיר את אי הווודאות בנוגע לבחירה עתידית כשהיה בזומת האב ולכן לא נבחר בחלוקת זו.

לאחר שהגדכנו בחלק זה מהו גרסור, נרצה להקדים הקדמה קצרה לנושא הספציפי בפרויקט שלנו ולימושים השונים אשר בהם השתמשנו.

לשם כך נרצה להציג מספר נקודות חשובות באשר למימוש חלק זה :

- עצי החלטה בהם נעשה שימוש למצאים תחת *Scikit – learn module* בשפת *Python*. העצים נבנים עצים ביןaries.
- נעשה שימוש רחב בשני מודלים. המודל הראשון הינו *DecisionTreeRegressor* המהווה גרסור המיזג עצ החלטה בודד. לעומתו משתמש בוסף ב- *RandomForestRegressor* אשר מאגד בתוכו מספר עצים החלטה.
- הפונקציה העיקרית בה נעשה שימוש הינה פונקציה מבנית המכונה "score". פונקציה זו מחשבת את ערך ה- *R squared* עבור גרסור. הסבר על פונקציה זו נכל בפרק הצגת מטריקות המדידה.
- קובוצת המבחן בה נעשה שימוש על-מנת לבחון את המודלים המוצגים לעיל הינה בת חמשה ימים. התאריכים בהם תחום ה- *data – set* שלו מובאים בפרק ה- *Preprocess* וקובוצת המבחן הינה חמשת הימים האחרונים של תאריכים אלה.

לאחר ההקדמה שזה עתה הציגנו נרצה לבנות גרסורים אשר יניבו עבורנו תוצאות טובות עד כמה שניתן לחיזוי מספר חוליות הקורונה ביישוב מסוים בתאריך ספציפי. לשם כך נתחיל את התהליך בהשוואה של המודלים בהם תחום *RandomForestRegressor* ו- *DecisionTreeRegressor* בחלק הבא של פרק זה.

חלק א' – עץ החלטה אל מול יער החלטה

בחלק זה נרצה לבצע השוואת השוואה בין *DecisionTreeRegressor* ל-*RandomForestRegressor* ראשית, נפרט בקצרה על כל אחד מרגורסוריים אלה:

- תוצאה מספרית מטווח רציף של מספרים. למשל חיזוי מספר חוליות קורונה בתאריך מסוים.
- אלגוריתם המבוסס על עצי החלטה כאשר כעת נלקחים בחשבון מספר עצים ולא עץ בודד. אלגוריתם זה למעשה דוגם בכל פעם דוגמאות שונות מתוך קבוצת האימון הנידתנת לו ובודנה עץ החלטה השונה מקודמו לאור הדוגמאות השונות הנבחנות בו. לאחר ייצרת כלל העצים, בהינתן אובייקט לסייע, אובייקט זה קיבל את הערך הממוצע של פלט כל העצים הנוצרים מה שיביל בתקווה לקבל חיזוי מדויק יותר עבור האובייקט הנבדק.

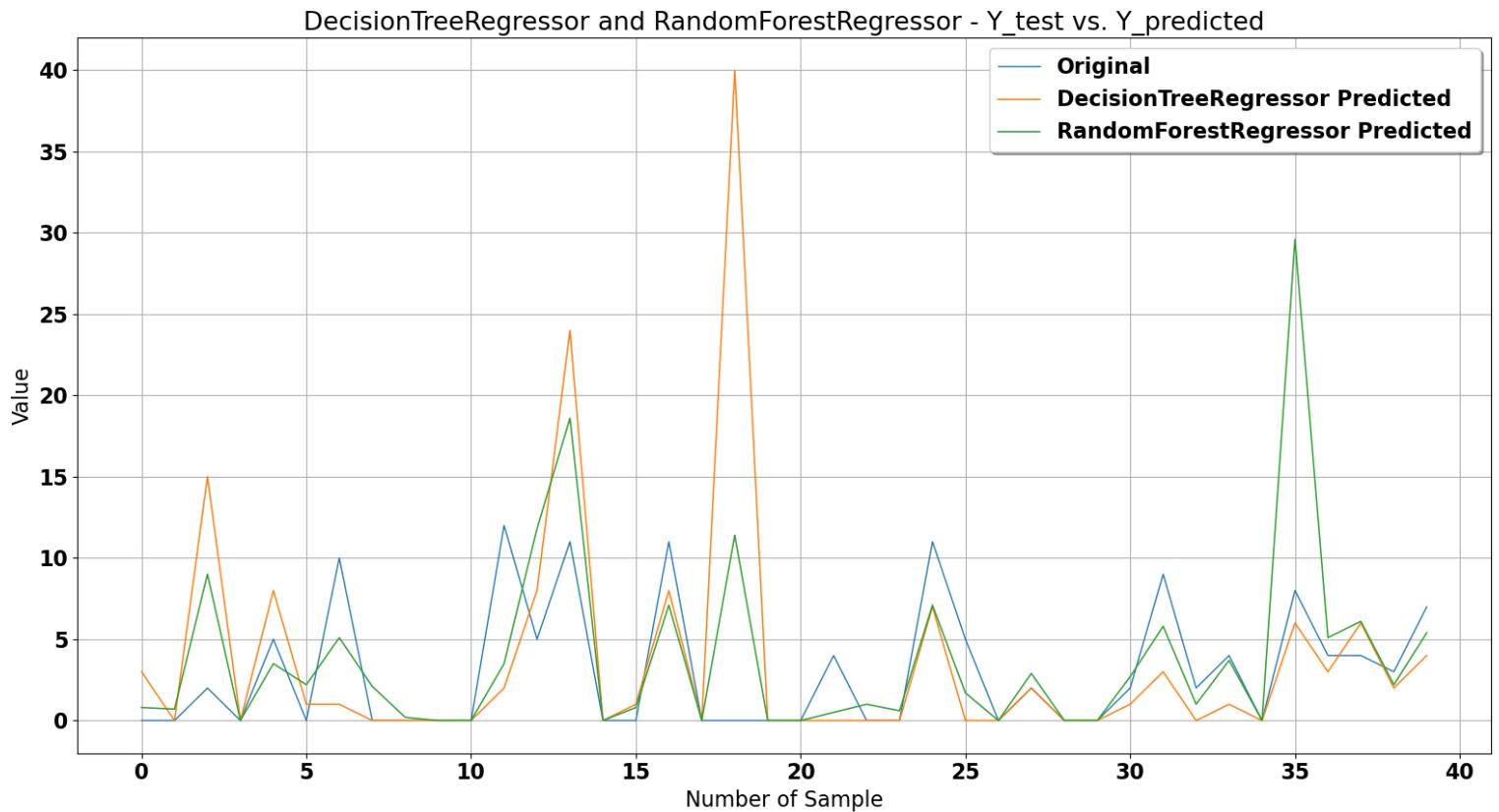
לאחר הפירוט על שני המודלים, נרצה לבצע השוואת ביניהם. כיוון ש-*DecisionTreeRegressor* מtabפס מעשה על עץ החלטה אחד ואילו *RandomForestRegressor* עושה שימוש במספר עצים החלטה לעומת מცפים לראות שיפור בתוצאות כאשר נרצה לקבל סיוג עבור דוגמא שתתחשב במספר עצים החלטה לעומת התคำשות בעץ ההחלטה בודד. נראה ניסוי שבו אנו מעוניינים לחזות חמישה ימים קדימה. ניסוי זה יהיה עבורנו נקודת מוצא שמננה נרצה לשפר את תוצאות המודלים במעלה הדריך. להלן תוצאות הניסוי:

```
DecisionTreeRegressor Test Set Score : 0.511  
RandomForestRegressor Test Set Score : 0.694
```

אכן, ניתן לראות בצילום המצורף כי חלה עלייה בפונקציית ה- "score" במעבר מ-*DecisionTreeRegressor* ל-*RandomForestRegressor*.

נציג גרפ' המראה את תוצאות הרוגסוריים על קבוצת המבחן. בגרף מוצגות תוצאותיהן של כ-40 דוגמאות מתוך קבוצת המבחן כאשר אנו מנסים לנבא חמישה ימים קדימה. עבור כל דוגמא, הגרף מראה את הערך האמתי שלה (*Original*) ואת הערכים אשר כל אחד מרגורסוריים העניק לה (*DecisionTreeRegressor Predicted/RandomForestRegressor Predicted*).

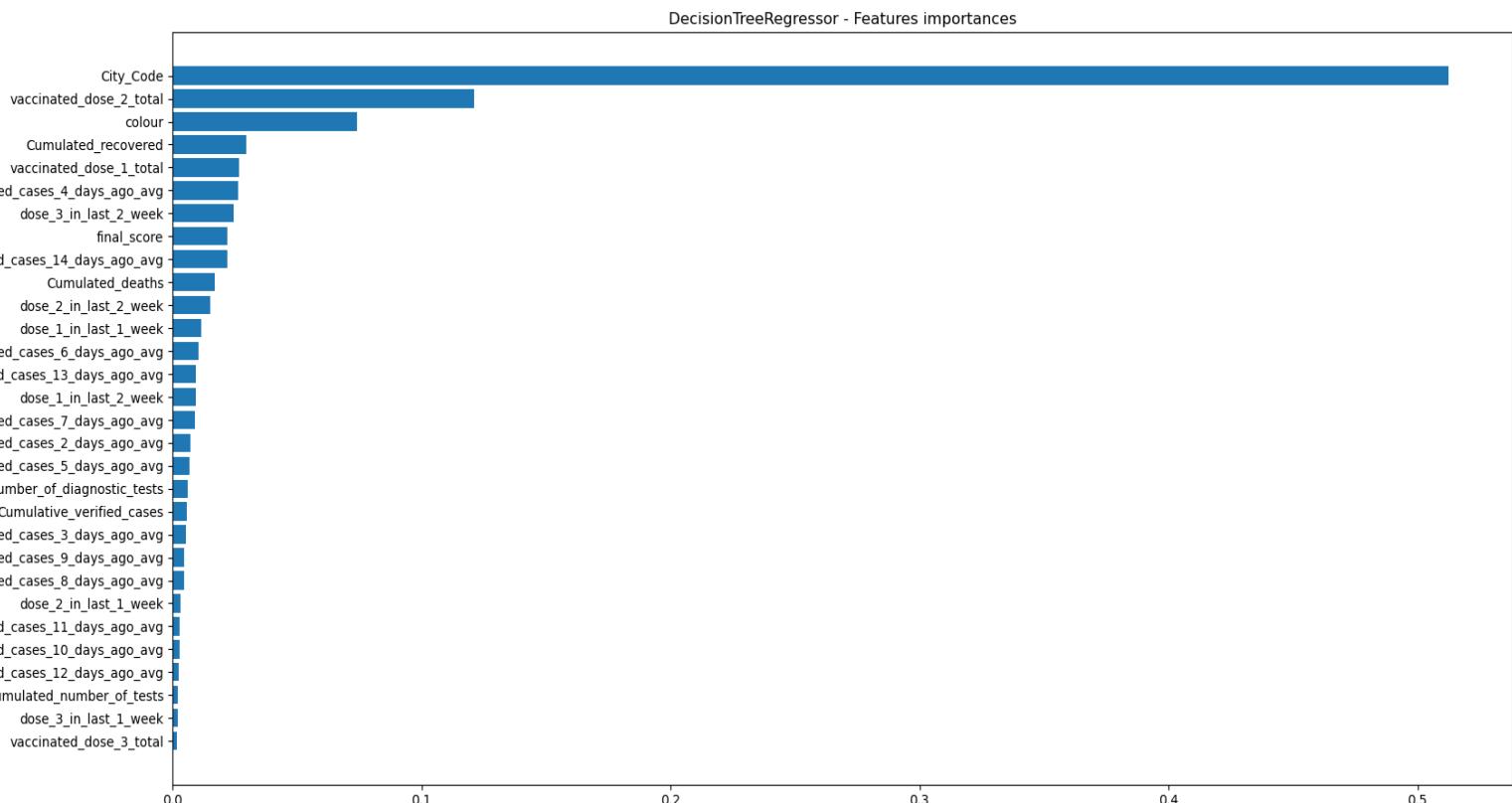
להלן הגרף:

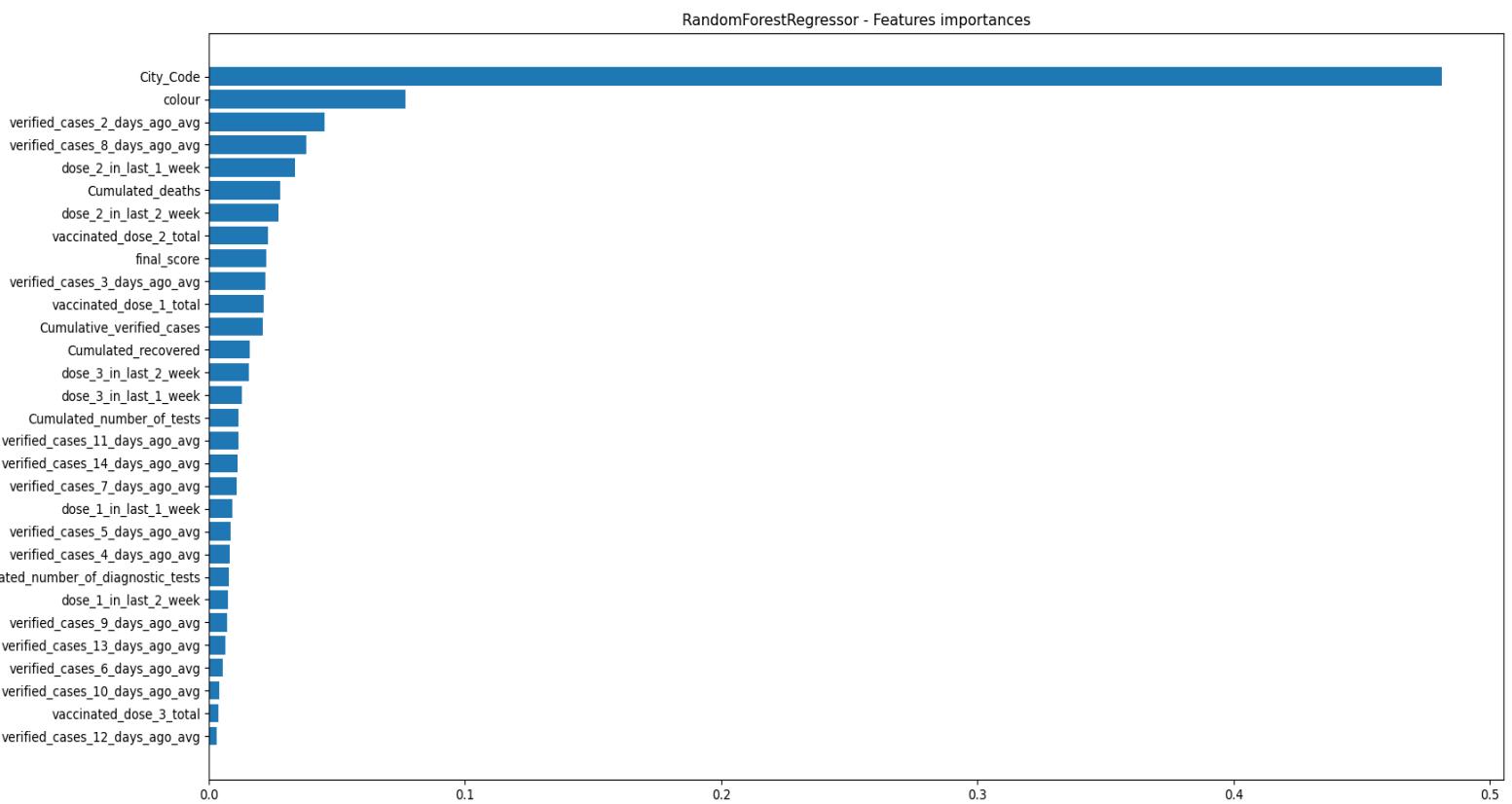


כעת, לאחר שהראינו תוצאה ראשונית (ע"ז בודד מול עיר), נרצה "לצלול" כעט לנושא הפרויקט שלנו ולנסות לקבל שיפורים נוספים בעזרת המודלים שלנו._CIDOU, וירוס הקורונה פגע בצורה קשה באספקטים שונים בחברה העולמית. לשם כך נרצה לשפר את תוצאות הרגרסורים שזהו עתה הפקנו על-מנת ליעיל את האלגוריתם שלנו ובכך להעלות את הדיוק בכל הקשור לחיזוי חוליה קורונה. על כך ב חלק הבא.

חלק ב' – בחירת תכונות רלוונטיות ביותר

כפי שתיארנו בסוף החלק הקודם, כעת נרצה לבנות רגרסור אשר ייב עבורנו שיפור בביצועים. לשם כך נדריך מושג **הטכונה feature importance**. מושג זה מתאר למעשה את רמת הרלוונטיות של הטכונה מסוימת במבנה הרגרסור/המסווג כאשר רמת רלוונטיות זו נמדדת באמצעות היכולת של הטכונה ליצור פיצול טוב יותר לשני צמותי בן בעת פיצולו של צומת אב ספציפי. למשל, במקרים אחרים, בהינתן רגרסור/מסווג ושתי תכונות f_1, f_2 , תכונה f_1 תיקרא רלוונטית יותר מטכנונה f_2 אם בבדיקה כלל הפיצולים האפשריים עבר כל צמותי העץ, תכונה f_1 הצליחה להשיג פיצולים טובים יותר מטכנונה f_2 . נראה כעת שני גרפים המראים את רמת הרלוונטיות של התכונות הנთונות בדוגמאות אשר הוזנו לרגרסורים. חשיבות כל טכונה תיוצג באמצעות ערך בין 0 ל-1 וסך כל חשיבותו התכונות נסכם ל-1. להלן הגרפים:





כפי שניתן לראות בצלומים המצורפים, רלוונטיות התכונה "City Code" הינה הגבוהה ביותר. נרים את ההשוואה בין *RandomForestRegressor* ל-*DecisionTreeRegressor* כפי שעשינו בחלק אי' כאשר בתחילת אנו מقلילים ב- "City Code" ובעמם השניה לא נכלל תכונה זו. להלן התוצאות:

```
DecisionTreeRegressor Test Set Score : 0.511
RandomForestRegressor Test Set Score : 0.694
```

Without City_Code Feature:

```
DecisionTreeRegressor Test Set Score : -1.147
RandomForestRegressor Test Set Score : 0.108
```

תוצאות אלה ממחישות באופן ישיר את חשיבות התכונה שהרוי כאשר הכלנו תכונה זו התקבלו רמות דיווק גבוהות יותר מאשר במקרה בו השמטנו תכונה זו. תוצאות אלה מראות התאמה עם האינטואיציה שלנו שהרוי "קוד היישוב" מהויה אינדייקציה מצוינת כאשר נרצה לסוג אובייקט בעל אותו "קוד יישוב". בambilים אחרות, כאשר נרצה לסוג אובייקט חדש, נרצה לבקר דוגמאות אשר מכילות את אותו "קוד יישוב" שהרוי מדבר באותו יישוב ולכן רמת הדמיון בין האובייקט לדוגמא תהיה גבוהה ביותר.

כעת, ננסה להתמודד עם השאלה הבאה. כאשר בנוינו את הרגורסורים בחלק א', נעשה שימוש ב-- *train set* המכיל תכונות המשותפות לכל הדוגמאות. ובכן נרצה לבדוק האם בהינתן סט של תכונות הניתנות לנו, יוכל להשמית תכונות בעלות רלוונטיות פחותה יותר אשר עלולות לפגוע לנו בביצועים ובכך לבנות רגסורים אשר יתבססו על תכונות שניבנו לנו תוצאות טובות יותר. שאלה זו עולה יפה לאחר שהראנו שאכן קיימות תכונות רלוונטיות יותר ורלוונטיות פחות.

לשם כך נרצה להציג אלגוריתם מבוסס לבחור עבורנו את התכונות הרלוונטיות ביותר הנקרה בקיצור "RFE" הינו אלגוריתם מפורסם לבחירת תכונות ספציפיות בעלות ערך גבוהה ביחס לשאר התכונות כאשר מטרתו הינה להסיר את התכונות הפחות רלוונטיות ולבנות מסובג/רגסור הנבנה מהתכונות הרלוונטיות ביותר. בכל בחירה של מספר תכונות מסוימים, האלגוריתם יבנה וגרסור מכלל התכונות הנתונות וייעניק לתכונות אלו דירוג לפי רמת הרלוונטיות שלהן. לאחר בניית הרגסור, האלגוריתם ישמש פעם אחר פעם את התכונות הפחות רלוונטיות וימשיך לבנות וגרסור חדש עם התכונות הנותרות. האלגוריתם י חוזור על תהליך זה שוב ושוב עד לקבלת מספר התכונות המבוקש אותו יש לספק לו כקלט. בסופו של תהליך התכונות הנבחרות תהיינה התכונות בעלות הרלוונטיות הגבוהה ביותר לפי מושג הרלוונטיות שהגדכנו לעיל. באמצעות אלגוריתם זה נבחן בכל איטציה מספר תכונות כתלות בפרמטר אותו נקבע באמצעות *n_features_to_select*, בניית וגרסור אשר יביא בחשבון את התכונות הנבחרות בלבד אשר נבחרו באמצעות ה- "RFE" ולאחר מכן נשמר את תוצאת הדיקוק שלו עבור ה- *test - set* שלנו הנמדד עבור חמישה ימים קדימה. בסיום ריצת האלגוריתם, נקבל וגרסור אשר מכיל בתוכו את התכונות הרלוונטיות ביותר אשר הניבו את הדיקוק הגבוה ביותר.

לשם הרצת ה- *RFE* נבצע ניסוי באופן הבא :

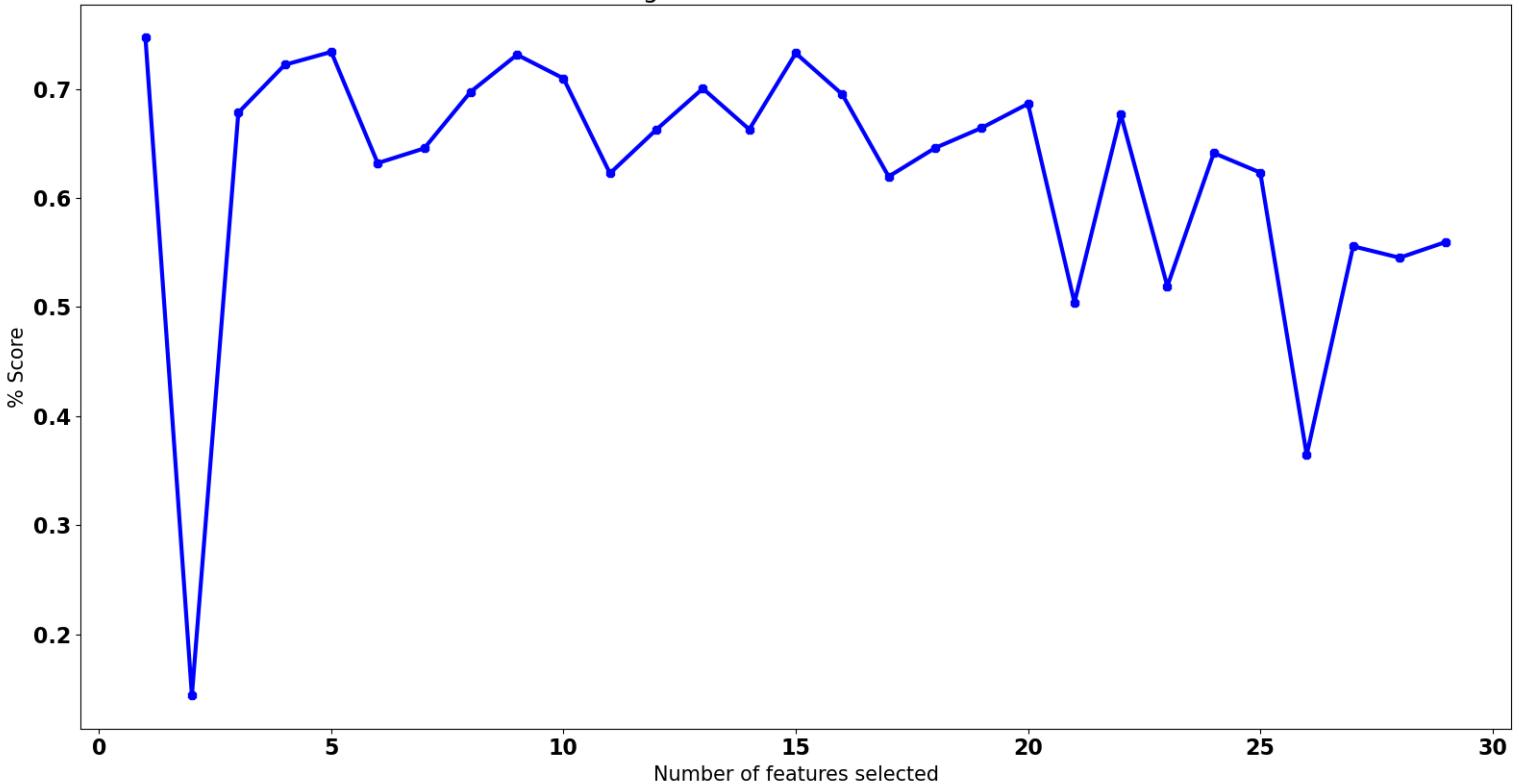
```
for featrues_to_select 1 to total_num_of_features:
```

```
run RFE(featrues_to_select)
```

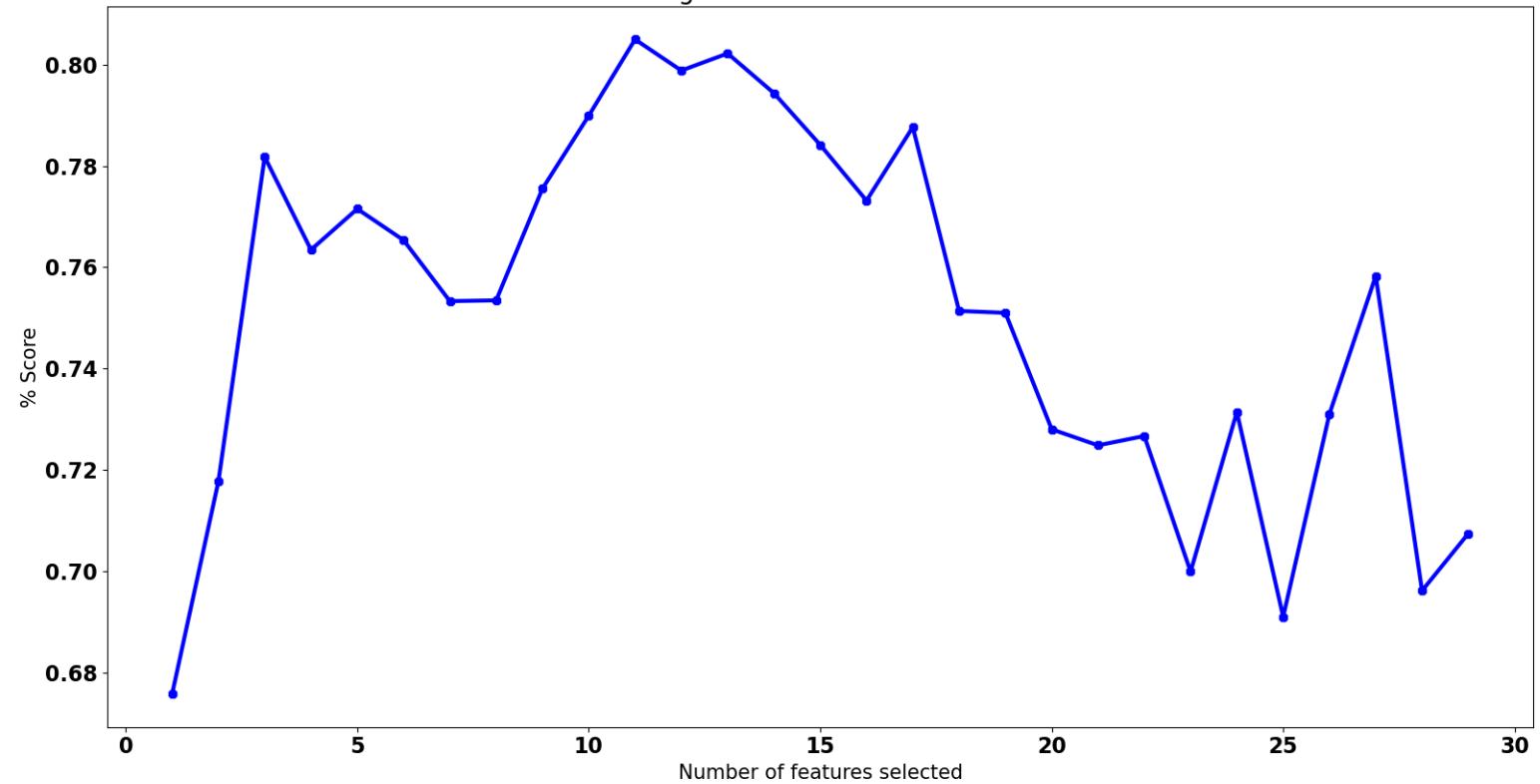
כעת נציג גرافים אשר מציגים את תוצאות האלגוריתם עבור כל אחד מהרגסורים. הגרף מתאר את רמת הדיקוק הגבוהה ביותר בהינתן מספר תכונות ספציפי אשר מהן נבנה הרגסור הטוב ביותר. תוצאות אלה התקבלו לאחר בניית המודלים באמצעות קבוצת האימון ומדידתם על קבוצת המבחן בגודל של חמישה ימים. הגרפים שנציג מראים את התוצאות עבור בחירה מטוחה של שתי תכונות ועד כלל התכונות הנתונות. גرافים אלה נבחרו להציג כיון שכאשר נבחרת תכמה אחת התקבלו תוצאות בעלות ערך שלילי גדול מה שגורם לחוסר המבשחה של שוני התוצאות.

להלן הגרפים:

DecisionTreeRegressor - Recursive Feature Elimination



RandomForestRegressor - Recursive Feature Elimination



כפי שניתן לראות קיימות שונות בין תוצאות הרגורסורים שנבנו כתלות במספר התוכנות שנבחרו. התוצאות הטובות ביותר ביותר מתקבליות כאשר נבחרו 2 תכונות עבור *DecisionTreeRegressor* ו- 12 תכונות עבור *RandomForestRegressor*. להלן התוצאות המתקבלות:

```
Regressor Type: DecisionTreeRegressor
Best Score: 0.747
Best Number Of Features: 2
Features Were Selected: ['City_Code' 'verified_cases_7_days_ago']

Regressor Type: RandomForestRegressor
Best Score: 0.805
Best Number Of Features: 12
Features Were Selected: ['City_Code' 'Cumulative_verified_cases' 'Cumulated_recovered'
'Cumulated_deaths' 'colour' 'final_score' 'vaccinated_dose_1_total'
'dose_2_in_last_2_week' 'dose_3_in_last_2_week'
'verified_cases_1_days_ago' 'verified_cases_6_days_ago'
'verified_cases_7_days_ago']
```

כפי שניתן לראות, אכן הצליחנו להשיג שיפור בכל אחד מהמודלים.

נרצה כעת לנשות המשיך ולשפר את הרגורסורים שלנו. לשם כך נציג אונגר בבנייה עצ-החלטה המכונה **התאמת-יתר** או בשם הלוואי ***OverFitting***. במיוחד, עץ החלטה נבנה באמצעות דוגמאות אלה הינו דוגמאות "רוועשות", כלומר ערוכה של התוכונה אותה אנו רוצחים לחזות לא מייצג ונונה את המציאות. בהינתן דוגמאות שכלה, יתכן מצב שבו הרצון להקטין את שגיאת האימון ככל האפשר (עד כדי שגיאת אימון אפס) עלולה לגרום לעלייה בשגיאת המבחן לאור העובדה שבבנייה העץ נעשתה עם רצון לבנותו תוך "התעקשות" ליצור הפרדה בין כלל הדוגמאות כולל הדוגמאות רוועשות. בכך להתגבר על בעיה זו נעשה שימוש במושג המכוונה **גיזום** לעץ / ***Pruning***. גיזום עץ ההחלטה יקטין את העץ ובכך יגדיל את שגיאת האימון תוך שאיפה ל渴בלת שגיאת מבחן קטנה יותר. כאשר נרצה לבצע גיזום לעץ החלטה, יהיה علينا לספק פרמטר ספציפי ש"יאמר" לאלגוריתם متى להפסיק את תהליכי בניית העץ. במקרה זה נרצה לבצע גיזום כאשר הפרמטר לגיזום יציג את מספר הדוגמאות המינימלי בצומת.

נרצה להריץ את האלגוריתם *RFE* בו השתמשנו בו לעיל לבחירת התכונות הטובות ביותר ביותר ובנוסף לבצע גיזום של העץ.

לשם כך נריץ את האלגוריתם עם גיזום הרגורסורים באופן הבא:

```
for samples_in_leaf in range [1 – 10]:
    for featrues_to_select 1 to total_num_of_features:
        run RFE(featrues_to_select,samples_in_leaf)
```

להלן התוצאות המתקבלות לאחר בניית המודלים באמצעות קבוצת האימון ומדידתם על קבוצת המבחן בגודל של חמישה ימים:

```

Regressor Type: DecisionTreeRegressor
Best Score: 0.777
Best Number Of Features: 10
Features Were Selected: ['City_Code' 'Cumulative_verified_cases' 'Cumulated_recovered'
'Cumulated_deaths' 'colour' 'final_score' 'vaccinated_dose_1_total'
'dose_3_in_last_2_week' 'verified_cases_7_days_ago'
'verified_cases_14_days_ago']
Min Samples in Leaf: 6

Regressor Type: RandomForestRegressor
Best Score: 0.805
Best Number Of Features: 12
Features Were Selected: ['City_Code' 'Cumulative_verified_cases' 'Cumulated_recovered'
'Cumulated_deaths' 'colour' 'final_score' 'vaccinated_dose_1_total'
'dose_2_in_last_2_week' 'dose_3_in_last_2_week'
'verified_cases_1_days_ago' 'verified_cases_6_days_ago'
'verified_cases_7_days_ago']
Min Samples in Leaf: 1

```

כפי שניתן לראות בתוצאות המוצגות, פועלות הגיזום הניביה תוצאות טובות יותר בעבר ה-*RandomForestRegressor* על-אף שעבור ה-*DecisionTreeRegressor* קיבלנו שלא ניזום התוצאה המתקבלת היא הטובה ביותר.

לאחר שהצגנו שיפור בתוצאות כאשר נבחרו תכונות ספציפיות לכל מודל, נרצה לנסות ולהתאים את התכונות שעליהם התאמנו המודלים לתכונות המתאימות יותר לכארה לבעיית *Time Series* אותה אנו מנסים להתמודד. על כך בחלק הבא.

חלק ג' – החלפת תכונות קיימות בתכונות חדשות

בالمשך כאמור לעיל בסוף חלק ב', נסעה לשפר את תוצאות הרגรสורים שלנו מעט יותר. לשם כך נציג את השאלה הבאה. כאשר אנו מעוניינים להתמודד עם בעיית *Time Series* ומנסים להתאים לרגסטור המוצג כע' החלטה, האם ישן תכונות שהיינו מעוניינים להוסיף לאובייקטיבים על-מנת לקבל תוצאות טובות יותר?

ובכן, נתבונן בתכונות הנבחרות כאשר ביצעו את הגיזום עבור שני המודלים. נשים לב שנבחרה תכונה משותפת 'City Code' שאת חשיבותה ראיינו לעיל. בנוסף נבחרה תכונה נוספת בשני המקרים והיא מספר המאומתים לפני שבעה ימים. דהיינו, המודלים שלנו הסתמכו על תכונה זו ובכך קיבלו תוצאות גבוהות יותר. כיוון שאנו למדים שמספר החולים החדשים לפני שבעה ימים אחורה על-מנת לנבא את מספר המאומתים ביום הנוכחי. על-מנת לקבל אינדיקציה טובה יותר על מספר החולים בעבר, נחליף את התכונות המייצגות את מספר המאומתים ב- X הימים האחרונים. נבצע ניסוי בו נתאמן על קבוצת האימון עם התכונות החדשות ונבצע גיזום המוצע ב- X הימים האחרונים. נבצע ניסוי בו נתאמן על קבוצת האימון עם התכונות החדשות ונבצע הרצאה: ובחירה תכונות כפי שעשינו בחלק ב'. להלן תוצאות הרצאה:

```
Regressor Type: DecisionTreeRegressor
Best Score:  0.75
Best Number Of Features:  9
Features Were Selected:  ['City_Code' 'Cumulated_recovered' 'Cumulated_deaths' 'colour'
 'final_score' 'vaccinated_dose_1_total' 'dose_3_in_last_2_week'
 'verified_cases_4_days_ago_avg' 'verified_cases_14_days_ago_avg']
Min Samples in Leaf:  10
```

```
Regressor Type: RandomForestRegressor
Best Score:  0.818
Best Number Of Features:  8
Features Were Selected:  ['City_Code' 'Cumulative_verified_cases' 'Cumulated_deaths' 'colour'
 'final_score' 'vaccinated_dose_1_total' 'verified_cases_2_days_ago_avg'
 'verified_cases_8_days_ago_avg']
Min Samples in Leaf:  8
```

נשים לב שעבור ה- *DecisionTreeRegressor* קיבלנו ירידה בדיק וailo עבור ה- *RandomForestRegressor* קיבלנו שיפור בתוצאות. ובכן ננסה לתת השערה מדוע לתופעה זו קורתה כאשר החלפנו את התכונות של מספר המאומתים ב- X הימים החולפים בתכונות חדשות במספר החוליםים המוצע ב- X הימים האחרונים. ובכן, כפי שהזכרנו לעיל, לפי תוצאות הניסוי בחלק ב' אנו למדים שהסתכלות של שבעה ימים אחורה מעניקה לנו אינדיקציה טובה כאשר אנו מעוניינים לבצע חיזוי על קבוצת המבחן. אמנם נשים לב שכאשר אנו בונים *DecisionTreeRegressor* אנו עושים שימוש בכל הדוגמאות בקבוצת האימון ולכן קיימת רציפות בתאריכים וailo כאשר אנו בונים יער החלטה באמצעות *RandomForestRegressor*, נבנים עצים שונים אשר כל אחד מהם נבנה מתוך-קבוצה של דוגמאות

מתוך קבוצת האימון. אי-כך אנו משערים כי ה"חורים" הנוצרים בקבוצת האימון כאשר אנו בונים עיר החלטה פוגעים ביכולת שלנו לבנות את קבוצת המבחן שהרי בעת אין רצף בתאריכים. כאשר החלפנו את התוכנות, אנו משערים כי התוכנות אשר מייצגות את ממוצע המאומתים בימים האחרונים הינן תוכנות פחות רלוונטיות (פחות חזקות) מאשר התוכנות המקוריות של מספר המאומתים האבסולוטי ולכן קיבלנו ירידה בדיקת אשר בנינו עצם החלטה בודד. אך לעומת זאת, כאשר נבנה עיר החלטה באמצעות התוכנות החדשות, אשר כל תכונה מייצגת בעת את ממוצע החוליםים בימים האחרונים, למרות שעיר בניו מעטים בעלי "חורים" בקבוצת האימון מהם הם נבנו, ממוצע החוליםים מצליח לחפות על-כך ולהציג תוצאה טובה יותר עבור עיר זה.

סיכום

כפי שהראנו בפרק זה קיימת חשיבות לתכונות בהן השתמשנו לצורך בניית הרגרסוריים. כאשר הרגרסוריים נבנו עם כל התכונות השגנו תוצאות טובות מאוד מאשר רגרסורים אשר נבנו עם תכונות ספציפיות בעלות ערך גבוה. לשם כך השתמשנו באלגוריתם "Recursive Feature Elimination" אשר בוחר את התכונות הרכלוונטיות ביותר ומהן בונה את הרגרסור בעל תוצאה ה-*score* הגבוהה ביותר. לשם שיפור נוסף השתמשנו בטכניקה המכונה "גיזום העץ" ואכן בעבר הרגרסור מסווג *DecisionTreeRegressor* הצלחנו לקבל שיפור ביצועים. לבסוף ביצעו החלפה של תכונות ובכך השגנו שיפור נוסף ביער החלטה שלנו.

נציג בטבלה את השיפורים לאורך הדרך :

Test – set: 5 days

<i>Regressors</i>	<i>Part A</i>	<i>Part B (No Pruning)</i>	<i>Part B (With Pruning)</i>	<i>Part C (With AVG)</i>
<i>DecisionTreeRegressor</i>	0.511	0.747	0.777	0.75
<i>RandomForestRegressor</i>	0.694	0.805	0.805	0.818

לסיום פרק זה, נציג את תוצאות הרגרסור הטוב ביותר שקיבלנו עבור מטריקות שונות כולל פונקציית ה-*score* בה עשינו שימוש לאורך הפרק. להלן התוצאות:

```
####--Metrics for RandomForestRegressor accuracy---###
Test Data Mean: 1.574
Mean Absolute Error: 1.186
Mean Squared Error: 7.169
Mean Absolute Percentage Error: 1441155189495168.5
Root Mean Squared Error: 2.678
Median Absolute Error: 0.419
R2 score: 0.818
```

Time Series Analysis

בחלק זה עוסוק בפתרון בעיית החיזוי באמצעות אלגוריתמי time series forecasting. מכיוון שהנושא שלא נלמד במסגרת קורס המבוא לבינה מלאכותית, תחילת נרצה לפתח את התאוריה מאחורי האלגוריתמי למידה המתאים לבעיות time series. משם נמשיך לבצע ניתוח מעמיק וטרנספורמציות על ה-data שלנו על מנת למצוא את המודלים המתאים ביותר לבעיית החיזוי שלנו. לבסוף נעבור להצגת המודלים לפתרון הבעיה, נבצע ניסויים ונציג אלגוריתמים ואופטימיזציות אשר יסייעו לנו בהשגת דיקט מרבבי בתחום החיזוי. עולם התוכן בו עוסוק כולל מונחים רבים וטרמינולוגיה עשירה. לכן, טרם נצלול אל תוך בעיית החיזוי שלנו נרצה לבצע סקירה של מספר מושגים וכליים מתמטיים אשר יעזור לנו למדל את הבעיה ולהגדיר את הטרמינולוגיה בה נשתמש לאורך הפרק.

רקע תיאורטי

נציג תחילת מספר מושגים בסיסיים ונגדיר את הטרמינולוגיה בה נשתמש בפרק הבא :

Observation (מדידה) – ערך כלשהו הנמדד בנקודת זמן כלשהי. הערך יכול לייצג מגוון של תכונות (טמפרטורה, משקל, אחז, מס' בדיד וכו') ונקודת הזמן יכולה להינתן ברמת גרגולריות משתנה (דקות/שעות/חודשים וכו'). לעיתים נשתמש גם במונח "דגם" בדומה חליפית (interchangeably). – סדרה של מדידות אשר מסודרות לפי סדר כרונולוגי בזמן. ההפרש בין כל שתי מדידות בסדרה הוא קבוע. בסדרת מדידות מסווג time series, הזמן הוא לרוב משתנה בלתי תלוי והמטרה היא לבצע חיזוי של מדידות עתידיות בהינתן הידע על המדידות שהיו בעבר. צורת חיזוי זו נקראת time series forecasting.

הערה : לאורך הפרק הבא יישתמש במונחים time series, data, data set בדומה חליפית.

Lag operator – נשתמש במונח lag כדי לבטא פעולה על שתי מדידות מרוחקות זו מזו מרחק k . המרחק נקבע לפי הגרגולריות של ה-time series שלנו. לדוגמה, עבור time series אשר בו נדemosות מדידות טמפרטורה יומיות, כאשר נאמר $lag 7$ הכוונה היא לפעולה (או אופרטור) בין מדידות אשר ההפרש ביניהן הוא בדיק שבוע.

תכונות מתמטיות וניתוח סטטיסטי

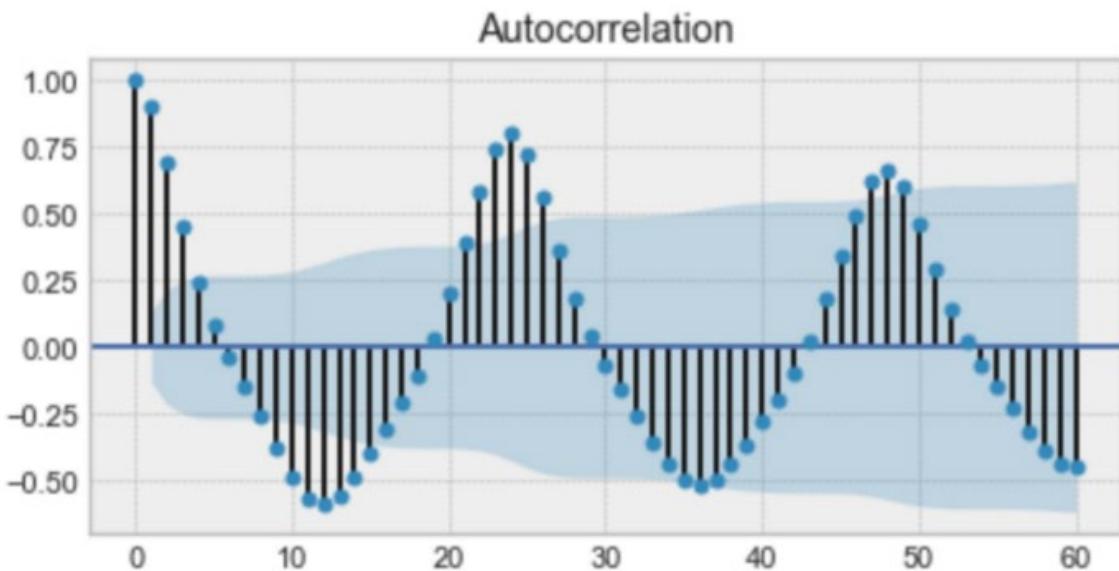
Autocorrelation function (ACF)

איןטואיטיבית, אוטו-קורלציה היא מدد לקרבה בין מדידות כפונקציה של המרחק ביניהן. פונקציה זו מאפשרת לנו לקבל ממדד המבטא את השפעה של מדידה בזמן t על מדידה בזמן $t + k$.

פורמלית – אם נסמן את ערך ה-time series בזמן t ב- x_t , פונקציית האוטו-קורלציה (Autocorrelation function) מניבה קורלציות בין x_t לבין x_{t-k} כאשר $\{0\} \cup \mathbb{N} \in k$ מסמל את ה-lag

$$\text{וformula הוא : } \rho_k = \frac{\text{cov}(x_t, x_{t-k})}{\sqrt{\text{var}(x_t) \cdot \text{var}(x_{t-k})}}$$

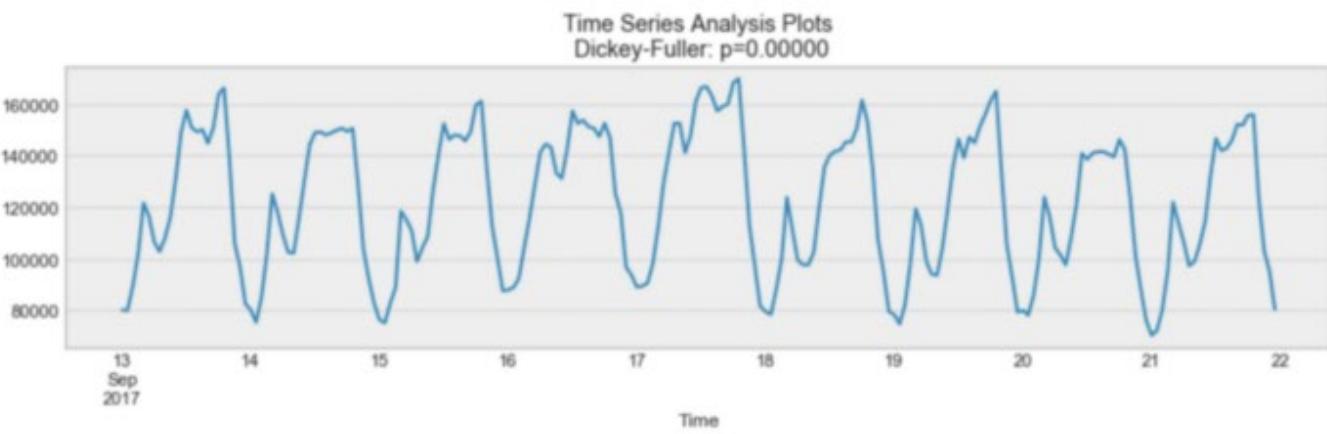
דוגמה לגרף Autocorrelation :



ניתן לראות בגרף כי בין הערך הראשון לערך ה-24 ישנה אוטו-קורלציה גבוהה. כמו כן גם בין הערך ה-12 לערך ה-36 ישנה אוטו-קורלציה גבוהה. מכך נוכל להסיק כי אנחנו יכולים למצוא ערכים דומים ב-time series שלנו עבור lag 24. בנוסף, הדפוס אותו אנחנו רואים מרמז לנו שקיימת תופעת עונתיות(time series) ב-time series שלנו.

מושג העונתיות (Seasonality) – מושג זה מתיחס לדפוסים חוזרים אשר מופיעים באורך lag-lags קבועים. לדוגמה, מכירות של גלידה יהיו תמיד גבוהות בקיץ ונדוכות בחורף, וצריכת חשמל תהיה לרוב גבוה בשעות היום ונמוכה בלילה.

נתבונן לדוגמה ב-time series המציג נתוני צריכה חשמל יומיים:



ניתן לראות כאן את דפוסי העונתיות – בכל יום ישנה עלייה בצריכת החשמל באמצעות היום וירידה בצריכה בשעות הערב והלילה.

מושג האינטיגרטיות (Volatility) – לרוב משתמשים במונח זה בחיזויים פיננסיים, על מנת לתאר את מידת אי הייציבות של מחיר מניה לאורך זמן. מזד זה נמדד לפי סטיית התקן של הרוחחים מהשעיה כלשהי. בקונטקט בו

אנחנו עוסקים, נשתמש במונח זה על מנת לתאר אי יציבות בסטיית התקן של הדגימות לאורך זמן. מבחינה אינטואיטיבית הכוונה היא שהגרף המתאר את time series שלנו לא "חלק".

Partial Autocorrelation function (PACF)

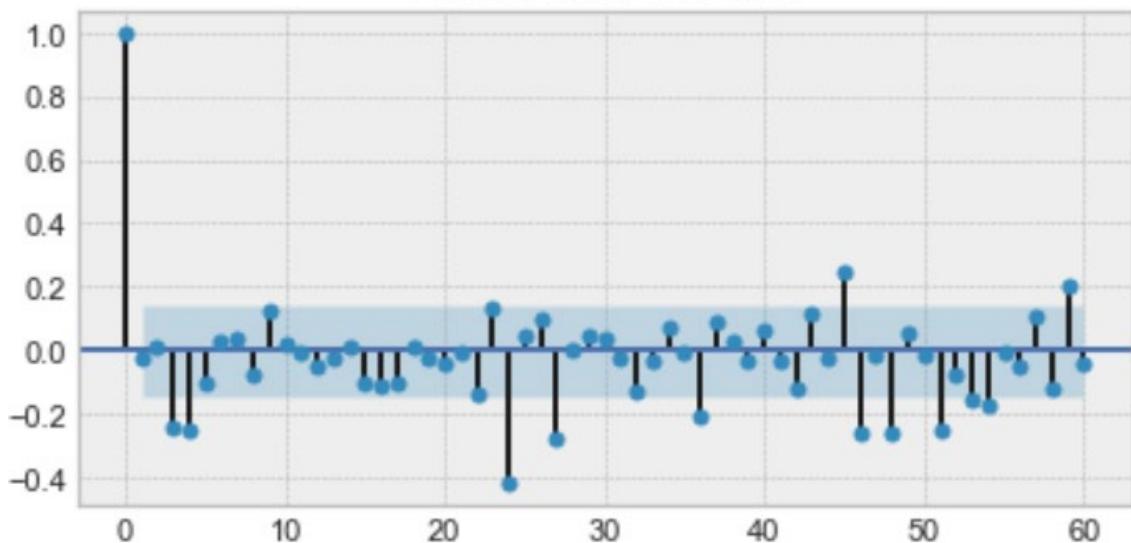
באופן דומה לפונקציית ACF, גם פונקציית PACF מודדת קורלציה בין מדידות שונות, אך הן נבדלות אחת מהשנייה בהבדל מהותי. נזכיר כי מטרתנו היא לקבל ממדד קורלציה בין מדידה x_t לבין מדידה x_{t-k} (lag k). ההבדל המהותי בין ACF ל-PACF הוא שפונקציית PACF מודדת את הקורלציה הישירה בין שתי המדידות, בעוד פונקציית ACF לוקחת בחשבון גם את ההשפעה של המדידות האחרות הנמצאות בטוחה שבין x_t ל- x_{t-k} . בכך אנחנו מקבלים קורלציה אשר מתחשבת **בمיקום** של המדידות בתחום ה-time series עצמו.

פורמלית – פונקציית PACF עבור זמן t ו- k תוגדר באופן הבא :

$$PACF(t, k) = \frac{cov(x_t, x_{t-k} | x_{t-1}, x_{t-2}, \dots, x_{t-k+1})}{\sqrt{var(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-k+1}) \cdot var(x_{t-k} | x_{t-1}, x_{t-2}, \dots, x_{t-k+1})}}$$

דוגמה לgraf PACF :

Partial Autocorrelation



הערה לגבי הגрафים – ניתן לראות כי בgraf ה-PACF כמו כן גם בgraf ה-ACF שהוצג לעיל ישנה יריעה כחולה המשתרעת לאורך ציר ה-X. ירעה זו מייצגת את הסוף הזניחה. המשמעות היא שכל הקורלציות אשר נמצאות **בתוך** תחומי היריעה הכחולה נחותות לוניות מבחן סטטיסטי ומשום כך ניתן להתעלם מהן (להחישbn כ-0), בעוד הקורלציות שנמצאות **מחוץ** לתחומי היריעה הכחולה הן קורלציות בעלות משמעות סטטיסטי לא זניחה, ולכן נוכל להתחשב בהן במודלים שלנו.

זהות תכונה חשובה מאד בה עליינו להתחשב כאשר אנחנו רוצחים להערך time series כלשהו. נאמר שההערך כלשהו הוא סטציונירי אם התכונות הסטטיסטיות שלו לא משתנות כתלות בזמן. פורמלית, הטעון סטציונירי הוא הטעון סטטיסטי אשר צפיפות הפילוג המשותפת שלו לא משתנה כתלות בזמן. עבור השימושים שלנו בפרויקט זה, על מנת לומר שההערך הוא סטציונירי, נדרש שיתקיים שלושה תנאים:

1. התוחלת μ קבועה בזמן.
2. סטיית התקן σ קבועה בזמן.
3. לא קיימת עונתיות (seasonality).

על מנת שנוכל לבצע חיזוי על ה-time series שלנו, נרצה שה-time series שלנו יקיים את תכונת ה-stationarity (עד כמה שאפשר). תנאי זה הוא תנאי הכרחי עבור כל אלגוריתמי ה-time series (עד כמה שהוא נ逋וק בפרק זה).

על מנת לקבל אינדיקציה האם ה-time series עליינו אנחנו עובדים מקיים את תכונת ה-stationarity, נוכל להיעזר במספר כלים:

1. ויזואלית – להציג את ערכי ה-time series ולבחון את התוחלת והמגמתיות של הנתונים. אמנם שיטה זו לא פורמלית, אך במקרים רבים זוהי השיטה המהירה והנוחה ביותר, ויכולת להביא לתוצאות דיווק טובות. שיטה זו נועה במיוחד מונת זהות דפוסי seasonality או שונות גדולה בתנאים.
2. מבחנים סטטיסטיים – על מנת לקבל אינדיקציה חד משמעית באשר למידת ה-stationarity של ה-time series שלנו, נוכל להפעיל מבחנים סטטיסטיים שונים אשר מספקים קритריונים מובחנים למידת ה-stationarity של ה-time series.

במסגרת פרויקט זה לא נרצה להעמיק בשילול המבחנים הסטטיסטיים השונים ובניהם מעמיק של תוצאות מבחנים אלו. נציג בקצרה את המודל התיאורטי אשר עומד מאחורי מבחנים אלו ונתאר את המדרדים בהם ניעזר על מנת להכריע באשר למידת הסטציונריות של ה-time series שלנו.

Statistical Hypothesis Testing

תחום זה בסטטיסטיקה עוסק באופן בו ניתן להגדיר השערות ולבוחן אותן על בסיס נתונים נצפים. נתחליל בהציג המושגים הכלליים:

- Hypothesis (היפותזה ראשית) – השערה או קביעה שברצוננו לבדוק.
- Null Hypothesis (היפותזה האפס) – זוהי היפותזה אשר מוגדרת כברירות מחדל. היפותזה זו היא ההיפותזה שכרגע מקובלת לפי דעת הרוב. לרוב מדובר בערך נאיבי כלשהו שנגזר מההיפותזה הראשית. נסמן ערך זה בתור H_0 .
- Alternative Hypothesis (היפותזה אלטרנטיבית) – ההיפותזה האלטרנטיבית (או היפותזה המחקר) היא היפותזה חלופית אשר מערבת את היפותזה הראשית. נסמן ערך זה בתור H_a .

ו- H_a הם הפכים מתמטיים. אנחנו יכולים להניח את H_0 עד אשר יוכח אחרת. במהלך תהליך הבדיקה שלנו ננסה לבדוק את ההיפותזה האלטרנטיבית. התוצאות האפשרות של תהליך זה:

1. לדוחות את H_0 – המשמעות היא שאנו חושבים ש- H_a היאבחירה טובה יותר.
2. להיכשל בדוחית H_0 – המשמעות היא ש- H_0 היא אכן הבחירה הטובה ביותר.

דוגמה: נניח כי אנחנו עובדים במפעל לייצור מטבעות. במפעל ישנה מכונה אשר מייצרת מטבעות, כאשר משקל של כל מטבע נקבע להיות 5 גרם.icut, אחד מהנדסי המפעל מגיע וטוען כי המכונה אינה מדיקת בחישוב המשקל, וכי המשקל האמיתי של המטבעות אינו 5 גרם. נגידר את המשתנים השונים עבור הבעיה הנתונה שלנו:

- היפותזה – ישנה שגיאה במשקל המטבעות המיוצרות ע"י המכונה.
- H_0 – משקל מטבע הוא 5 גרם.
- H_a – משקל מטבע אינו 5 גרם.

כפי שניתן לראות, H_0 ו- H_a הם הפכים מתמטיים.

icut, על מנת לבדוק את ההיפותזה האלטרנטיבית, תיכנן מהנדס ניסוי בו דוגם באופן אקראי 100 מטבעות אשר המכונה מייצרת. המהנדס שקל את כל אחת מהמטבעות ובחן את התוצאות שקיבל. התהליך שביצע המהנדס נקרא **test statistic**.

Test Statistic – סטטיסטיקה המוחשבת על פי קבוצת מדוגם כלשהי בגודל קבוע הנלקחת מתוך הנקודות. התוצאה מייצגת רזונציה של הנקודות לערך נומרי ייחד אשר אותו ניתן להחלטת אם לקבל או לדוחות את ההיפותזה האלטרנטיבית.

בדוגמה לעיל, test statistic עבור הניסוי של המהנדס הוא המשקל הממוצע של מטבע על סמך הדגימות שלקח בניסוי שלו. בדינה של הערך הממוצע תעזר מהנדס להחלטת האם ההיפותזה האלטרנטיבית שהציג מספיק חזקה על מנת לדוחות את H_0 . לדוגמה: אם המשקל הממוצע המתתקבל מהתוצאות הניסוי הוא 6.43 גרם, ניתן לדוחות את H_0 ולקבל את H_a . מנגד, אם המשקל הממוצע המתתקבל הוא 5.04 גרם, לא ניתן לדוחות את H_0 מכיוון שהערך הממוצע מציבע על כך שעבור רוב המוחלת של הדגימות המשקל אכן היה 5 גרם.

אנחנו זוקקים למדוד אשר יעזור לנו לבדוק את ה-**test statistic** שלנו ולהחליט אם ניתן לדוחות את H_0 . לשם כך נשתמש בהסתברות הנקראת **p-value**.

נגידר את המושגים הרלוונטיים:

- **Level of confidence** – ערך זה מייצג עד כמה אנחנו בטוחים בבחירה שלנו. לרוב נקבע את ערך זה להיות בין 95% – 99%. נסמן ערך זה בתורו c .
- **Level of significance** – נסמן ערך זה באות α וערך זה: $c = 1 - \alpha$.
- **p-value** – ערך זה הוא קונספט סטטיסטי אשר משתמשים בו בתחוםים רבים בעולם – science והבינה המלאכותית, החול מ-**hypothesis testing** ועד לעצם החלטה, ריגרסיות ועוד. ככל

שערך ה- p יותר גבוהה, כך גובר הסיכוי ש- H_0 נכונה, וככל שה- p -value נמוך כך גובר הסיכוי לדחות את H_0 . ערך p -value הסטנדרטי בו משתמש עבור המבחןים הסטטיסטיים שלנו שווה ל- α (level of significance). בקונטקט זה, הערך c מבטא את ההסתברות שההינתן דוגמה מתוך קבוצת המבחן שלנו, נקבל ערך אשר דוחה את H_0 . באופן משלים, ערך ה- p -value (α) היא ההסתברות לקבל דוגמה אשר לא דוחה את H_0 .

נזור אל הדוגמה שלנו – נניח שערך ה- p המתאים הינו $0.05 = \alpha$. המשמעות היא שעבור 5% מהמקרים בניסוי התקבל משקל השווה בדיקות 5 גרם, או באופן שקול שעבור 95% מהדוגמאות מתקבל ערך השונה מ-5 גרם. על פי התוצאות האלה ניתן לקבוע בביטחון כי הדמייה של H_0 היא לגיטימית, וכי ניתן כעת לקבל את H_a בתור היפותזה עדיפה.

כלל האכבע בו נשתמש עבור ערך p -value הוא **של ערך מתחת ל-5%** יהו אישור לקבלת H_a .

נזור כעת לבעה המקורית שהיא אנחנו מתמודדים – בהינתן time series, נרצה לסוג האם הוא מקיים את תוכנות הסטציונריות. על מנת לבצע את הסיווג נשתמש בetest סטטיסטי שנקרא (Augmented ADF). בetest זה, היפותזת האפס שלנו H_0 time series שהוא לא סטציוני, והhypothese האלטרנטיבית H_a שהוא time series שה- p -value של מנת לקבוע האם ה- p -value להריץ את המבחן הסטטיסטי ADF ולהשתמש בערך ה- p -value על מנת לקבוע האם time series סטציוני או לא.

```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -3.421990
P-Value                  0.010232
# Lags Used              19.000000
# Observations Used     630.000000
Critical Value (1%)      -3.440772
Critical Value (5%)      -2.866139
Critical Value (10%)     -2.569219
dtype: float64
Is the time series stationary? True
```

דוגמה לפטל של המבחן הסטטיסטי ADF על time series נניתן להלן:

כללי כלשהו :
- ניתן לראות כי הפטל מכיל את ערך test statistic שמניב ה-ADF, בנוסף לערך p -value, מספר הדגימות עליה בוצע המבחן הסטטיסטי וערכים נוספים אשר נתונים אינדיקטיביים למידת החשיבות הסטטיסטית של הערכים.

הערות :

- בחרנו שלא להרחיב על התיאוריה המתמטית מאחוריו ה-ADF test. למטרת הפרויקט שלנו המידע שהוৎגן כאן מספק כדי לקבל את הרקע התיאורטי והאינטואיציה העומדת מאחוריו השימוש במבחן זה.
- ישנו קרייטריונים רבים המשמשים להערכת מידת הסטציונריות של time series נתון. עבור השימושים שלנו במסגרת פרויקט זה נסתפק בהערכת ערך p -value כמדד לסטציונריות, וכך בغالל שערך ה- p נמוך מ-5% יוכל לסוג את time series כסטציוני.

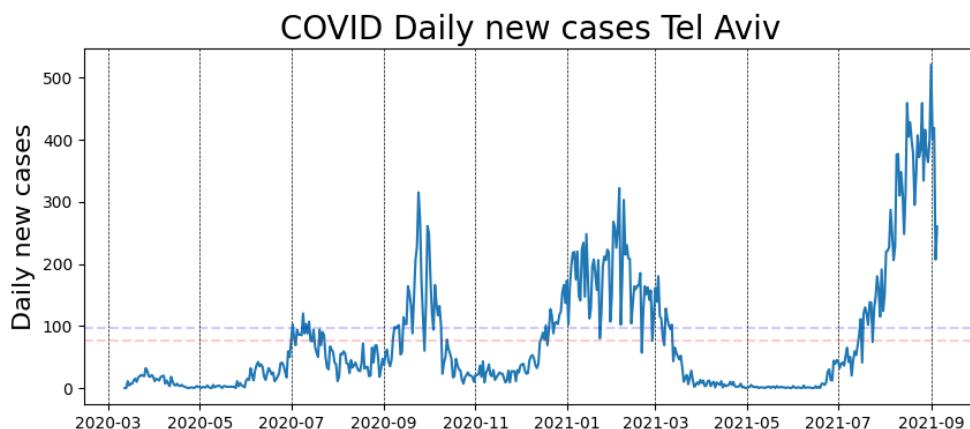
בחים האמיטיים פעמים רבות נתקל ב-time series stationary שאינו stationary. במקרים אלו יוכל לבצע transformation על מנת להפוך את ה-time series לstationarity.

Time Series Transformations

בاهינתן time series אשר לא מקיים את תכונת הסטציונריות, נרצה להפוך אותו לסטציונרי כדי שנוכל להשתמש באלגוריתמי הלמידה שלנו לחיזוי של ערכים עתידיים. על מנת להפוך את הנתונים לסטציונרים, ניתן להפעיל טרנספורמציות על ערכי המדיות ב-time series שלנו. כל טרנספורמציה היא למעשה פונקציה אשר אנחנו מפעילים על כל ערכי המדיות. הטרנספורמציה שאנו מפעילים צריכה להיות הפיכה, כדי שנוכל תמיד לחזור אל הערכים המקוריים אותם אנחנו רוצים בסופו של דבר לחזות. בנוסף, כדאי לרוב להשתמש בטרנספורמציות פשוטות עם סיבוכיות חישוב נמוכה, כל עוד הן מצלילות להשיג את המטרה.

נתאר כאן מספר טרנספורמציות שניתן להפעיל על ה-time series שלנו. לצורך הדגמה של השפעת הטרנספורמציות נסתכל על time series ספציפי, נבצע עליו טרנספורמציות ונראה את השפעת הטרנספורמציות על מידת הסטציונריות שלו.

ה-time series עליו נסתכל מתוך אט מספר המאומתים היומי בעיר תל אביב, בתקופת התאריכים שבין ה- 12/03/2020 ל- 09/05/2021. נציג את הנתונים בגרף :



מקרה עבור הגרפים אשר מוצגים בחלק זה :

- הקו האדום המוקווקו מציג את ערך התוחלת של כל הדוגמאות.
- הקו הכהול המוקווקו מציג את ערך סטיית התקן של כל הדוגמאות.

ניתן לראות מבחן ויזואלית של הגרף כי ישנה וולטיליות (אי יציבות) של הערכים סביב התוחלת וסטיית התקן. נರיץ את ה-test ADF על מנת לבדוק בצורה פורמלית את מידת הסטציונריות של הדטה :

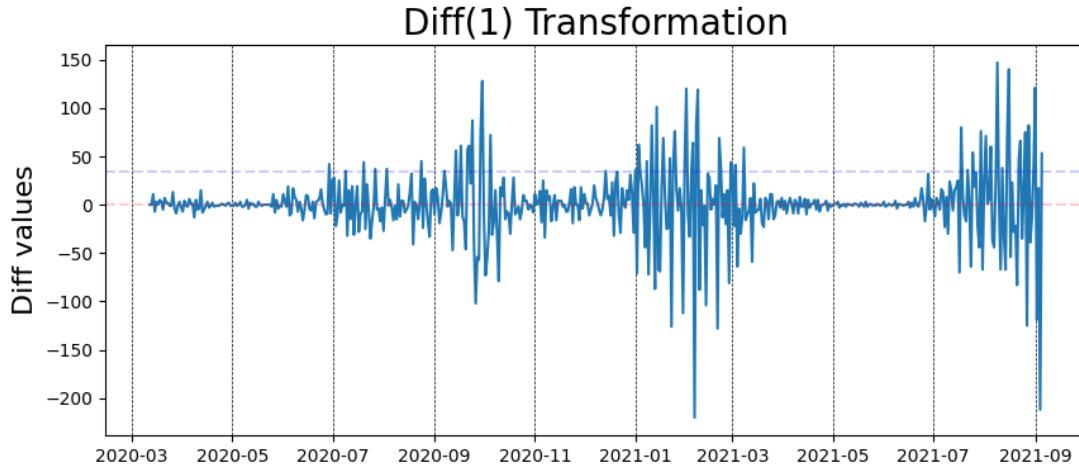
```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -2.639292
P-Value                  0.085145
# Lags Used              16.000000
# Observations Used     526.000000
Critical Value (1%)      -3.442843
Critical Value (5%)      -2.867050
Critical Value (10%)     -2.569705
dtype: float64
Is the time series stationary? False
```

כפי שניתנו לראות, ערך $\text{h}-\text{p}$ -value מ-8% ועל כן ה-time series שלנו אינו סטציונרי. נבחן בעת מספר טרנספורמציות שונות שניתן להפעיל על הדטה ונבחן כל טרנספורמציה כדי לראות אם היא מביאה את הדטה שלנו למצב סטציונרי.

בתיאור הטרנספורמציות השונות נשתמש בסימנו x_t על מנת לתאר את ערך הדגימה המקורי בזמן t ונשתמש בערך ϕ לסימנו ערך הטרנספורמציה בזמן t .

- $diff(k) -$ פונקציה זו מחליפה כל ערך בהפרש בין הדגימה בזמן t לבין הדגימה בזמן $t - k$.

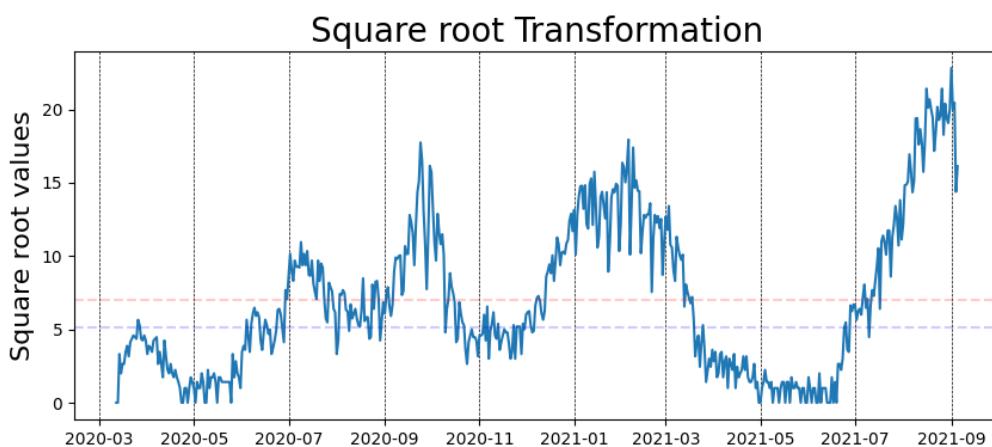
פורמלית: $\phi_t = x_t - x_{t-k}$. לאחר הפעלת הטרנספורמציה (1) על ה-*time series* שלנו נקבל:



```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -4.760588
P-Value                  0.0000065
# Lags Used              19.0000000
# Observations Used     523.0000000
Critical Value (1%)      -3.442915
Critical Value (5%)       -2.867082
Critical Value (10%)      -2.569722
dtype: float64
Is the time series stationary? True
```

ניתן לראות כי כעת התוחלת נשארת קבועה יחסית לזמן. מהפעלת ה-*ADF test* ניתן לראות כי כעת ה-*time series* שלנו סטציוני, כנדרש. בנוסף נשים לב כאשר אנחנו משתמשים בטרנספורמציה זו, ישנים פחות ערכאים בסדרה החדשה, וזאת משום שהערכים הראשונים ב-*time series* שלנו כעת לא יהיו ולידיים ולכן נזינח אותם, וכן סדרת הדגימות תתחילה מהיום הראשון עבורה יש לנו ערך תקין $diff$.

- הפעלת פונקציית שורש: $\sqrt{x_t} = \phi_t$. לאחר הפעלת הטרנספורמציה על ה-*time series* שלנו נקבל:

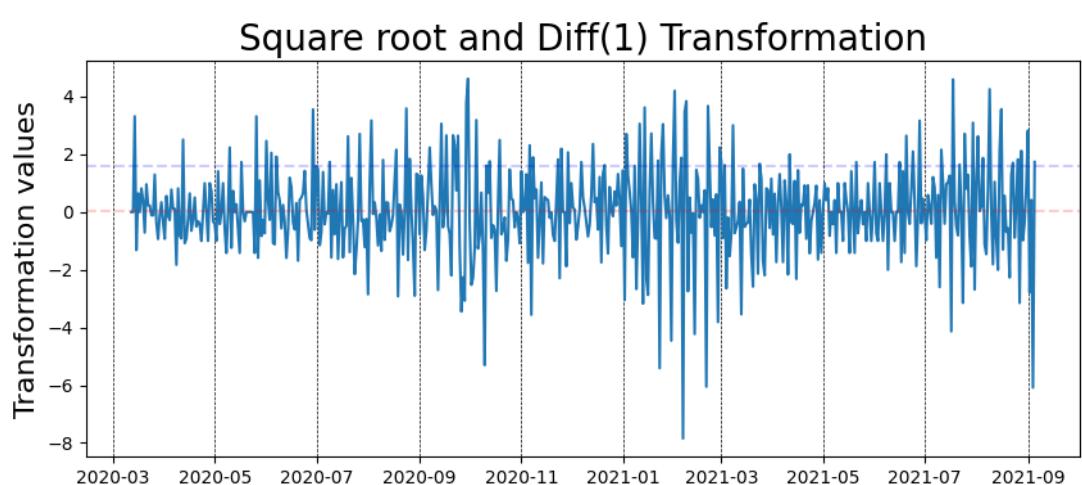


```

Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -2.262448
P-Value                  0.184365
# Lags Used             14.000000
# Observations Used     528.000000
Critical Value (1%)     -3.442796
Critical Value (5%)      2.867030
Critical Value (10%)    -2.569694
dtype: float64
Is the time series stationary? False

```

ניתן לראות מבחן ויזואלית כי הפעלת פונקציית שורש לא שינתה את המגמה הכללית של הנתונים. אם נפעיל את ה- ADF test נקבל את התוצאות המוצגות מושمال:
 ניתן לראות כי הפעלת הטרנספורמציה לא הצליחה להפוך את ה- time series שלנו לסטצionario, ואף הביאה לכך שערך ה- p-value עליה בהשוואה ל- time series המקורי שלו. במקרים בהם הפעלת הטרנספורמציה ייחידה לא מביאה לתוצאה המבוקשת, ניתן לבצע הרכבה של הטרנספורמציות. לדוגמה, במקרה זה נוכל לחת את הפלט של הטרנספורמציה השורש, ולהפעיל עלייה את הטרנספורמציה $\text{diff}(1)$. ככלומר הטרנספורמציה החדשה שלנו היא: $\phi_t = \sqrt{x_t} - \sqrt{x_{t-1}}$:



```

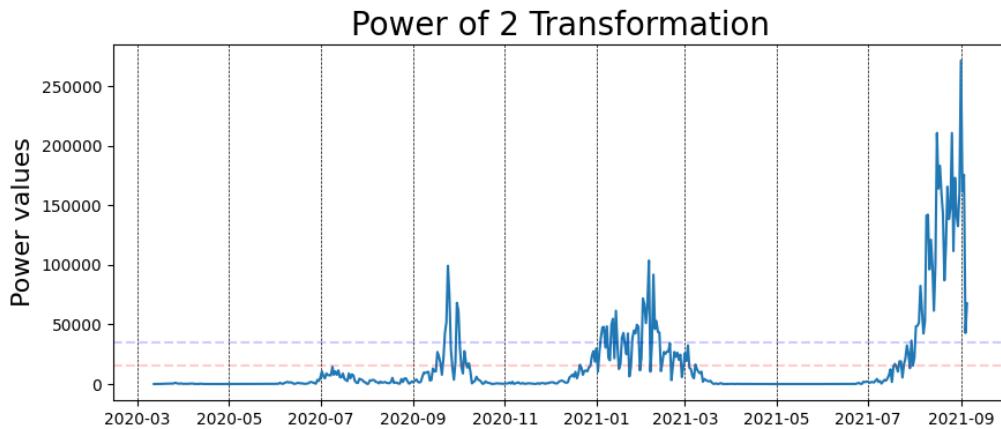
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -4.701090
P-Value                  0.000084
# Lags Used             19.000000
# Observations Used     523.000000
Critical Value (1%)     -3.442915
Critical Value (5%)      2.867082
Critical Value (10%)    -2.569722
dtype: float64
Is the time series stationary? True

```

כעת קיבלנו תהליך שנראה מבחן ויזואלית סטצionario. נפעיל את ה- ADF test ונקבל:

ניתן לראות כי כעת אכן קיבלנו תהליך סטצionario. בנוסף, חשוב לציין כי עליינו לבצע אך ורק הטרנספורמציות הפיכות. על מנת להפוך את הטרנספורמציה נוכל להפעיל את הטרנספורמציות ההופכיות בסדר הפוך לסדר הפעלה המקורי. במקרה זה מתקיים: $x_t = (\phi_t + \sqrt{x_{t-1}})^2$.

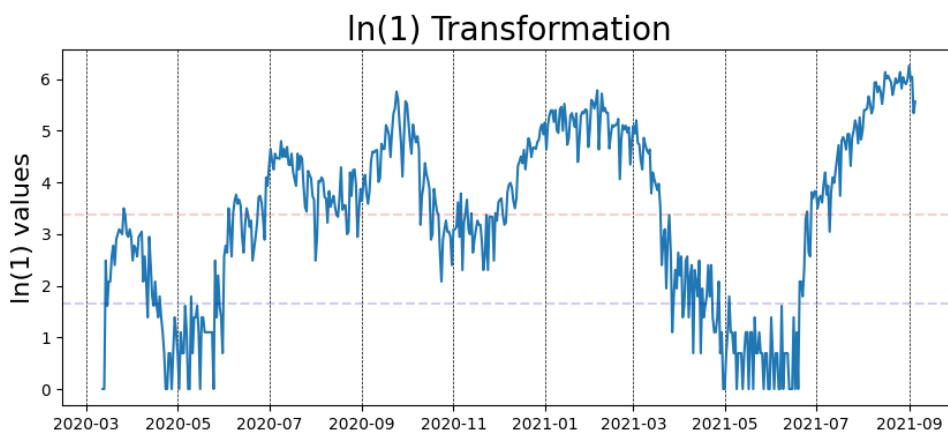
- x_t^q – העלאה בחזקה : ϕ . לאחר הפעלת הטרנספורמציה על ה time series שלנו קיבל :



```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -3.092112
P-Value                 0.027142
# Lags Used            16.000000
# Observations Used    526.000000
Critical Value (1%)     -3.442843
Critical Value (5%)      -2.867050
Critical Value (10%)     -2.569705
dtype: float64
Is the time series stationary? True
```

למרות ש מבחינה ויזואלית אולי קשה לראות זאת, הפעלת ה- ADF test תגלה לנו שהתהליך אכן סטציונירי לאחר הפעלת הטרנספורמציה. זהה דוגמה לכך שלפעמים ההערכה הויזואלית יכולה להטעות. עם זאת, תחילה זה עדין עשוי להיות קשה לחיזוי בשל הולטיות הגובהה שלו.

- $\ln(\Delta)$ – פונקציית הלוגריתם הטבעי. הארגומנט Δ מצין ערך אותו נוסיף לכל דוגימה מקורית, על מנת שנוכל להפעיל את פונקציית ה- \ln על ערכים גדולים ממש מ-0. פורמלית נגידר : $\exp(\Delta)$. הטרנספורמציה ההופוכה תהיה $\exp(\Delta) = \ln(x_t + \Delta)$ והיא תוגדר באופן הבא : $\phi_t = \ln(x_t + \Delta)$. לאחר הפעלת הטרנספורמציה (1) על ה time series שלנו נקבל :



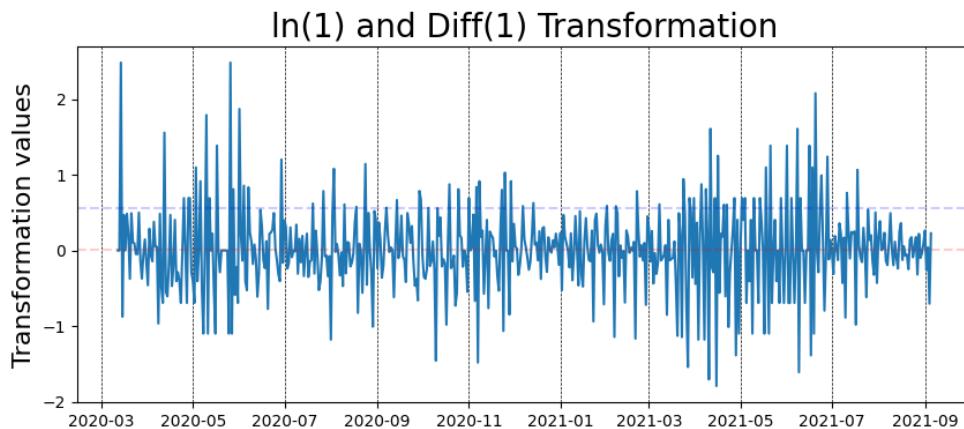
```

Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -1.912304
P-Value                  0.326295
# Lags Used             14.000000
# Observations Used     528.000000
Critical Value (1%)     -3.442796
Critical Value (5%)      2.867030
Critical Value (10%)    -2.569694
dtype: float64
Is the time series stationary? False

```

כמו בדוגמאות הקודמות, גם כאן ניתן לראות שהפעלת הטרנספורמציה לא הצליחה להפוך את התוחלת וסטיית התקן לקבועה לאורך זמן. תוצאות ה-ADF מראות את האבחנה הווייזואלית.

פעם נוספת ננסה לבצע הרכבה של טרנספורמציות על ידי הפעלת $\text{diff}(1)$ על תוצאה הטרנספורמציה הראשונה. תוצאה הרכבה:



כעת קיבלנו תהליכי סטציונירי עם p -value נמוך מאד.

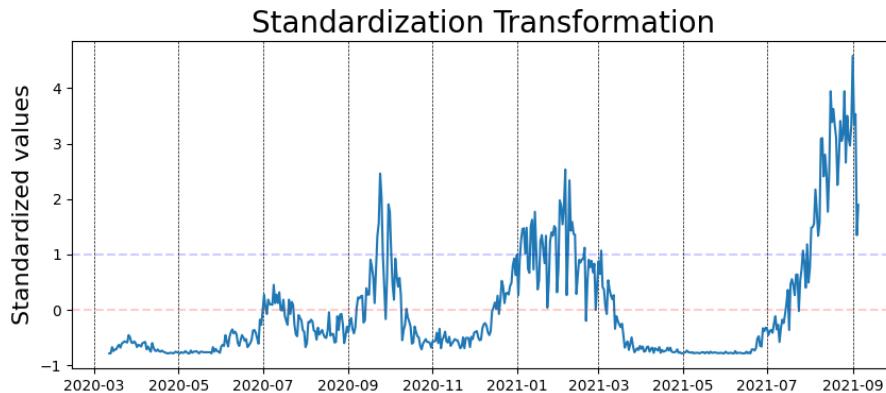
```

Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -5.302359
P-Value                  0.000005
# Lags Used             13.000000
# Observations Used     529.000000
Critical Value (1%)     -3.442772
Critical Value (5%)      2.867019
Critical Value (10%)    -2.569688
dtype: float64
Is the time series stationary? True

```

- standard_score – טרנספורמציה זו מבצעת נרמול של התהליך בהתאם לתוחלת וסטיית התקן.

פורמלית נגדיר: $\phi_t = \frac{x_t - \mu}{\sigma}$ כאשר μ היא התוחלת ו- σ סטיית התקן. לאחר הפעלת הטרנספורמציה על ה-time series שלנו קיבל:

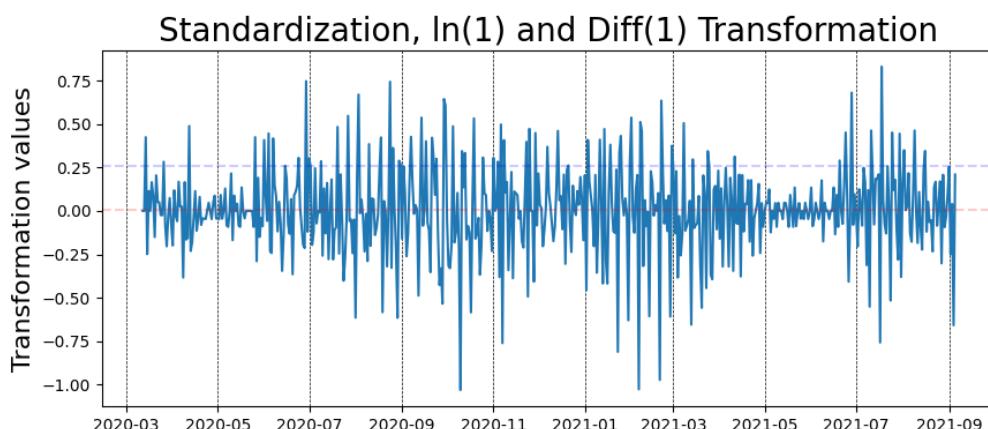


```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -2.639292
P-Value                 0.085145
# Lags Used             16.000000
# Observations Used     526.000000
Critical Value (1%)     -3.442843
Critical Value (5%)     -2.867050
Critical Value (10%)    -2.569705
dtype: float64
Is the time series stationary? False
```

באופן לא מפתיע, צורת הגרף נשארה זהה לחולtin, אך התוחלת וסטיית התקן של התהיליךicut קטנים הרבה יותר, וכל הערכים נמצאים בטווח שבין -1 – 5 . נספה שהטהיליך עדין לא יהיה סטציוני מכיוון שלא שינוו את התפלגות הערכים עצם, ואכן ניתן לראות כי תוצאות ה-ADF test זהות לאלו שהפענו על ה-time series המקורי.

icut נקבע הרכבה של טרנספורמציות על גבי טרנספורמציה-standard_score – תחילת נפעיל את הטרנספורמציה `ln(1)` ואחריה את הטרנספורמציה `.diff(1)`. הטרנספורמציה המורכבת נתונה ע"י:

$$\phi_t = \ln\left(\frac{x_t - \mu}{\sigma} + 1\right) - \ln\left(\frac{x_{t-1} - \mu}{\sigma} + 1\right)$$



```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -4.783899
P-Value                 0.000058
# Lags Used             19.000000
# Observations Used     523.000000
Critical Value (1%)     -3.442915
Critical Value (5%)     -2.867082
Critical Value (10%)    -2.569722
dtype: float64
Is the time series stationary? True
```

icut קיבלנו תhilיך סטציוני עם `p-value` נמוך מאד.

בחלק זה נוכחנו בגלות עד כמה תהליך ה-preprocessing שאנו מבצעים על ה-time series שלנו הוא חשוב ועלול להשפיע בצורה משמעותית על תוצאות החיזוי שלנו בפרק הבא. ראיינו כי בעזרת הבנה של המושגים הסטטיסטיים הבסיסיים ניתן למדוד הרבה על האופי של הדגימות שלנו, וכי ניתן לקחת תהליך שאינו סטטיאוני ולהפעיל עליו טרנספורמציות פשוטות אשר הופכות אותו לתהליך אופטימלי לחיזוי. בפרק הבא נראה כיצד ניתן להשתמש במושגים טרנספורמציות אשר יעוזו מנת לנתח את הדגימות אותן אנחנו רוצים לחזות, ובמידת הצורך נפעיל טרנספורמציות אשר יעוזו לנו להכין את הדגימות לתהליך החיזוי.

Time Series Forecasting

נרצה לעבור כעט לפתרון בעיתת החיזוי באמצעות אלגוריתמי time series. לפני שניגש לתיאור האלגוריתמים בהם נשתמש לפתרון הבעיה, נרצה למדל את ה-data שלנו כך שייתאים לפתרון של בעיות time series.

Data Preprocessing

בפרק ה-Data מידלנו את הנתונים שלנו כך שייתאים לאלגוריתמי למידה קלאסיים (KNN, DT). עבור אלגוריתמים אלו בנינו וקטור של תכונות אשר מהווים את הקלט למסווג. כאשר אנחנו רוצים לפתרון בעיתת time series קלאסית, התכוונה היחידה בה נרצה להתחשב היא התוכנה אותה אנחנו רוצים לחזות. בקרה שלנו, בעיתת החיזוי המרכזית שלנו מתרכזת בחיזוי של מס' המאומתים היומי בكورونا. כמובן, ה-data שלנו כולל רשימה כרונולוגית של תאריכים, כאשר עברו כל תאריך רשאי את מס' המאומתים היומי.

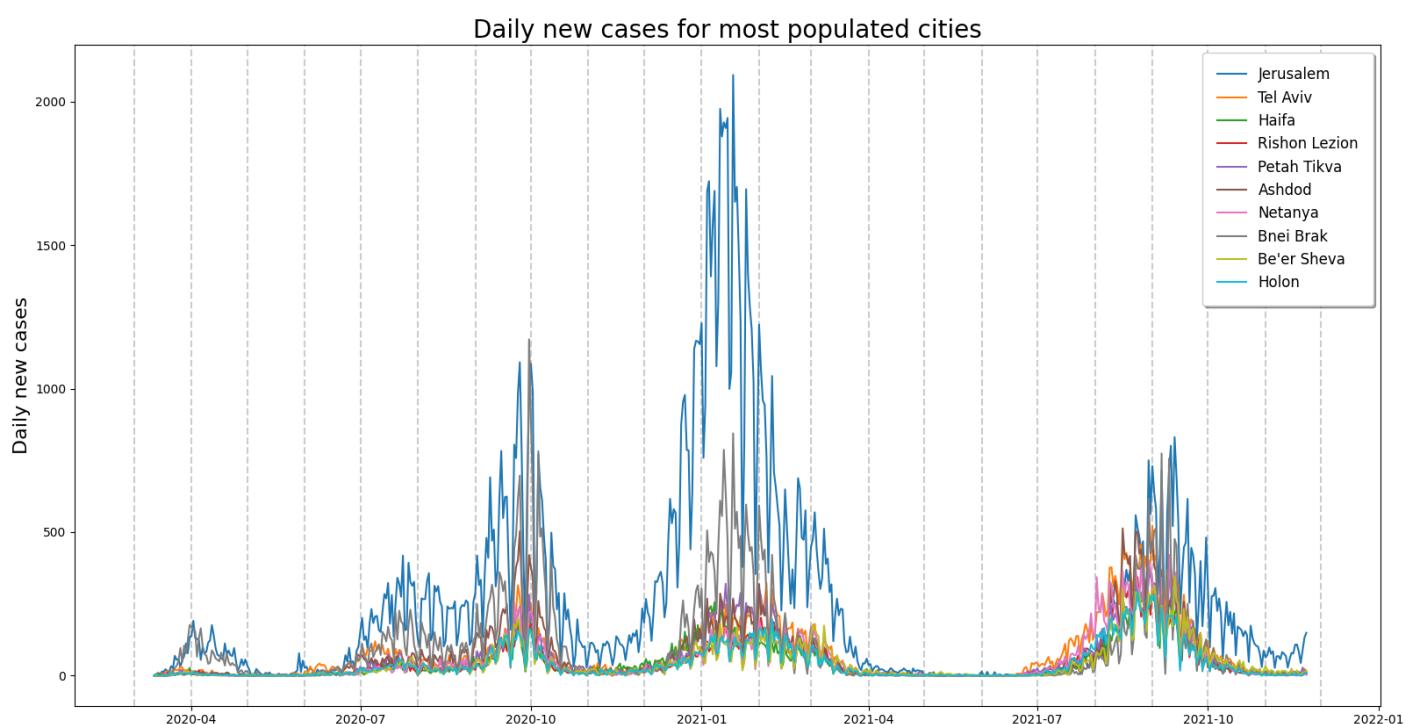
נרצה להפיק שני סוגי נתונים של time series datasets:

1. מס' מאומתים יומי לפי עיר.

2. מס' מאומתים יומי ברמה הלאומית.

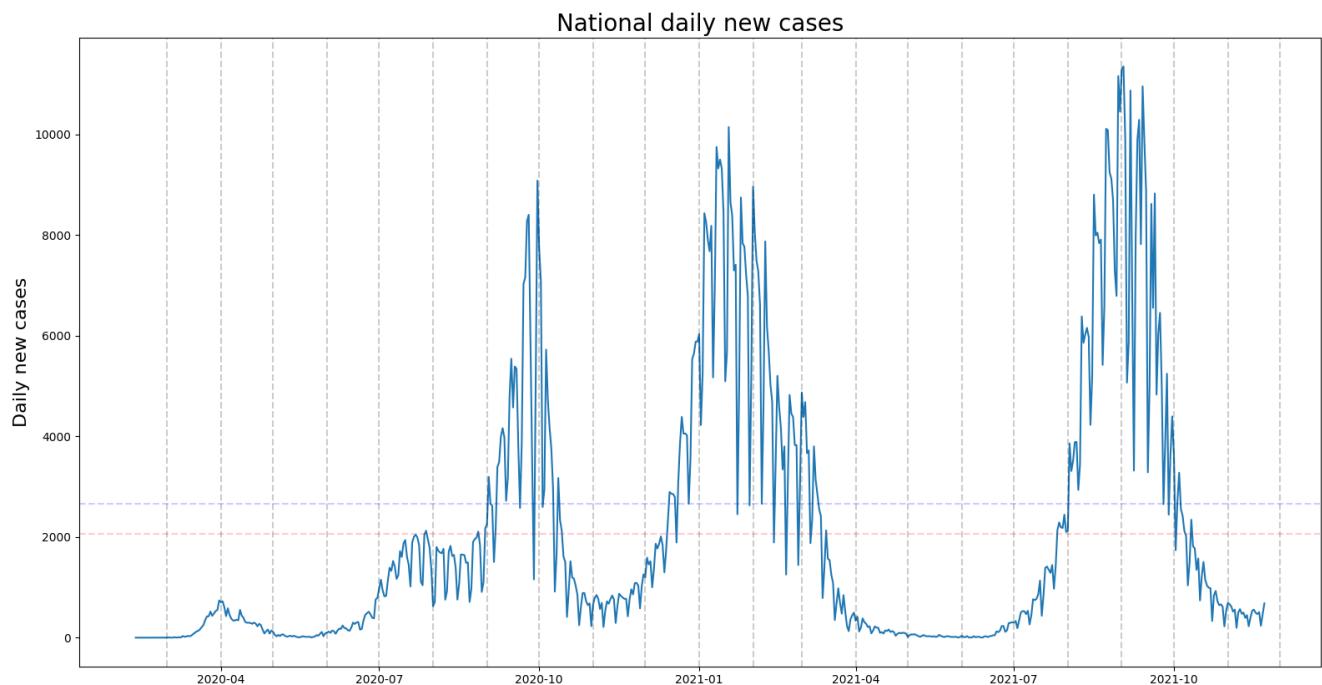
את כל המודלים והאלגוריתמים שנציג בפרק זה ניתן להריץ גם על מס' המאומתים היומי לפי עיר וגם על מס' המאומתים ברמה הלאומית.

נציג מס' פלטים לדוגמה של ה-preprocessing time series datasets. לאחר ביצוע תהליך preprocessing הנדרש. נציג את התוצאות בגרפים המתארים את מס' המאומתים היומי כתלות בתאריך. הגרף הבא מתאר את מס' המאומתים היומי לפי עיר, עברו עשרה הערים עם האוכלוסייה הגדולה ביותר:



עבור סדרות time series המופקotas לכל עיר, טווח התאריכים עליו יש לנו data הוא מה-12/03/2020 ועד 23/11/2021.

כעת נציג את הגרף המציג את מס' המאומתים היומי ברמה הלאומית :



מקרה :

- הקו האדום המקווקו מציג את ערך התוחלת של כל הדגימות.
- הקו הכהול המקווקו מציג את ערך סטיית התקן של כל הדגימות.

עבור סדרת time series המציגת את מס' המאומתים היומי ברמה הלאומית, טווח התאריכים עליו יש לנו data הוא מה-12/02/2020 ועד ה-22/11/2021. **מטרתנו בפרק זה תהייה לחזות את מס' המאומתים ברמה הלאומית.**

כעת נתחיל לתאר את אלגוריתמי הלמידה והחיזוי בהם השתמש על מנת לפתור את בעיית החיזוי שלנו. האלגוריתמים בהם נתמקד הם אלגוריתמי רגרסיה ליניאריים (linear regression model). במודלים אלו, הפרמטר אותו אנחנו חוזים יהיה מורכב מצירוף לינארי של פרדיקטים (predictors) שונים המורכבים מה-data שידוע לנו.

האלגוריתמים בהם עוסיק בפרק זה :

1. Moving average (MA)
2. Auto regression (AR)
3. Auto regression moving average (ARMA)
4. Auto regression integrated moving average (ARIMA)
5. Seasonal auto regression integrated moving average (SARIMA)

Moving Average Model

מודל ה-univariate moving average (בקיצור – MA) הוא מודל שימושי לפתרון תהליכיים מסווג series, הכולמר תהליכיים המכילים משתנה בודד. משתנה הפלט במודל (הערך אותו אנחנו רוצים לחזות) תלוי לינארית בשגיאות סטוכסטיות המגיעות מתוך התפלגות ידועה. נתאר את המודל בצורה פורמלית ולאחר מכן נסביר את האינטואיציה העומדת מאחוריו. מודל $MA(q)$ מוגדר באופן הבא:

$$y_t = \mu + \varepsilon_t + \phi_1 \varepsilon_{t-1} + \cdots + \phi_q \varepsilon_{t-q}$$

כאשר μ היא התוחלת של סדרת הדגימות, ϕ_1, \dots, ϕ_q הם פרמטרים של המודל, ו- $\varepsilon_t, \varepsilon_{t-1}, \dots, \varepsilon_{t-q}$ הם פרמטרי השגיאה או "הרעש" (white noise error). פרמטרים אלו מפולגים נורמלית $\varepsilon \sim N(\mu_\varepsilon, \sigma_\varepsilon^2)$. ברוב המודלים מתקיים $\varepsilon_i \sim N(0, \sigma_\varepsilon^2)$.

מהגדרת המודל ניתן לראות כי ערך של מדידה בזמן t שווה לערך התוחלת של הדגימות, בתוספת תיקוני שגיאות מהמדידות הקודמות. כדי להסביר את האינטואיציה מאחוריה המודל ניתן לחשב על \hat{y}_t ועל שגיאות ביחס לתוחלת. בכל שלב המודל מסתכל על השגיאות שהיו לו עבור הדגימות הקודמות בסדרה, ומנסה להקטין את השגיאה עבור החיזוי של הדגימה הבאה.

משוואת החיזוי עבור $MA(q)$:

$$\hat{y}_t = c + \phi_1 \varepsilon_{t-1} + \cdots + \phi_q \varepsilon_{t-q}$$

הפרמטר c של משוואת החיזוי מתאר את המקדם החופשי (constant) של המודל. עבור חיזוי מושלם יתקיים $\mu = c$. השגיאה של משוואת החיזוי נתונה ע"י: $\varepsilon_t = y_t - \hat{y}_t$, כאשר \hat{y}_t הוא ערך החיזוי ו- y_t הוא ערך האמת.

מציאת הפרמטר q עבור מודל MA

נרצה למצוא את הפרמטר q האופטימלי עבור המודל שלנו. לשם כך ניעזר בפונקציית ה-ACF (Autocorrelation). **תוצאות:** אם נסמן את ערך ה- x_t time series בזמן t ב- x_t , פונקציית האוטו-קורלציה (ACF-Autocorrelation function) מוגדרת כפונקציה קורלציונית בין x_t לבין x_{t-k} כאשר $k \in \{0\} \cup \mathbb{N}$ מסמל את ה-lag וערכה הוא:

$$cov(x_t, x_{t-k}) / \sqrt{var(x_t) \cdot var(x_{t-k})} = \rho_k.$$

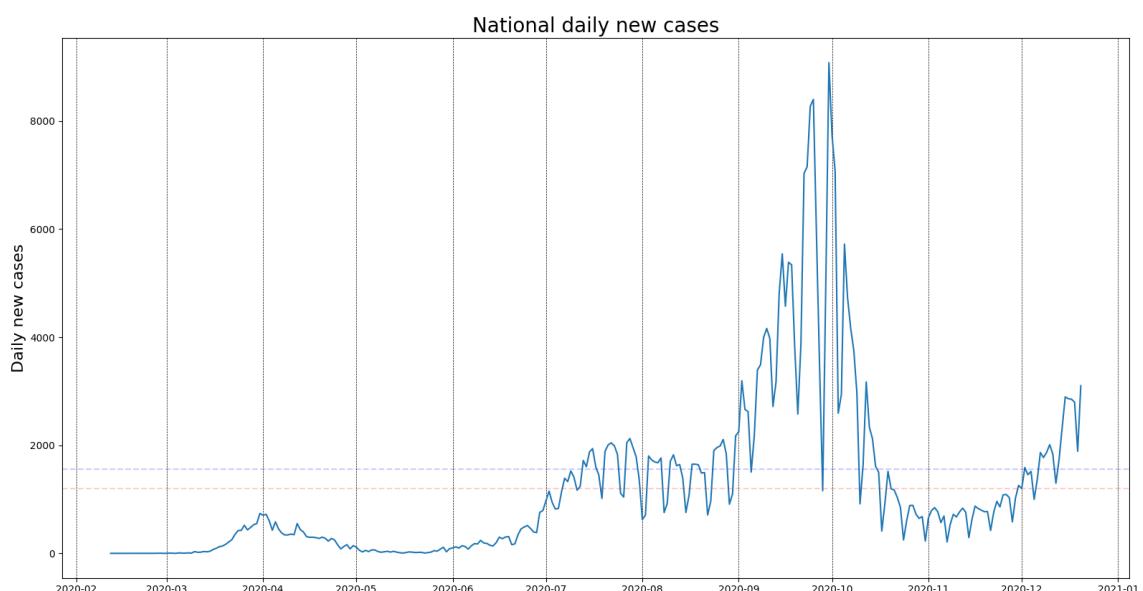
נינתן להוכיח באופן פורמלי כי $\rho_k \neq 0 \Leftrightarrow k \leq q$. גראן. לכן, נרצה לבחור את q כך שיכיל רק ערכים לא זניחים של ρ_k . לשם כך נוכל להיעזר בגרף ה-ACF. זה גם יכול לתת לנו אינדיקציה לעוונתיות שקיימת לנו ב-time series, ולדפסי הקורלציה השונים כתלות ב-lag שנבחר. קיימים tradeoff בבחירה הערך q – מצד אחד נרצה שהחיזוי שלנו יהיה מבוסס על כמה שיותר מידע מימיים קודמים, אך מצד שני התבססות על יותר מדי lags יכולה לגרום ל-overfitting של המודל שלנו ובנוסף עלול להביא לכך שנבנਸ את החיזוי שלנו על lags עבורם הקורלציונות עם הערך אותו אנחנו רוצים לנבא לא טובות.

חיזוי באמצעות ($MA(q)$)

לאחר בחירת הפרמטר q עבור המודל, יבוצע תהליך למידה בו יבחרו הפרמטרים עבור משווהת החיזוי. moving average לאחר מכן נוכל לבחור את התקופה אותה נרצה לחזות. יש לשים לב למוגבלת של מודל ה- $MA(q)$ – אם נבחר לחזות מספר ימים n כך ש- $n > q$, תוצאת החיזוי תהיה לכל הימים החל מהיום $n - 1 + q$, ותהיה שווה לערך התוחלת של משווהת המודל.

נראה כעת דוגמה בסיסית לחיזוי באמצעות מודל moving average. נראה את שלביים השונים החל מטיעינת ה-data, אימון המודל וניתוח התוצאות. לצורך הדוגמה נשתמש ב-time series המציג את מס' המאומתים ברמה הלאומית.

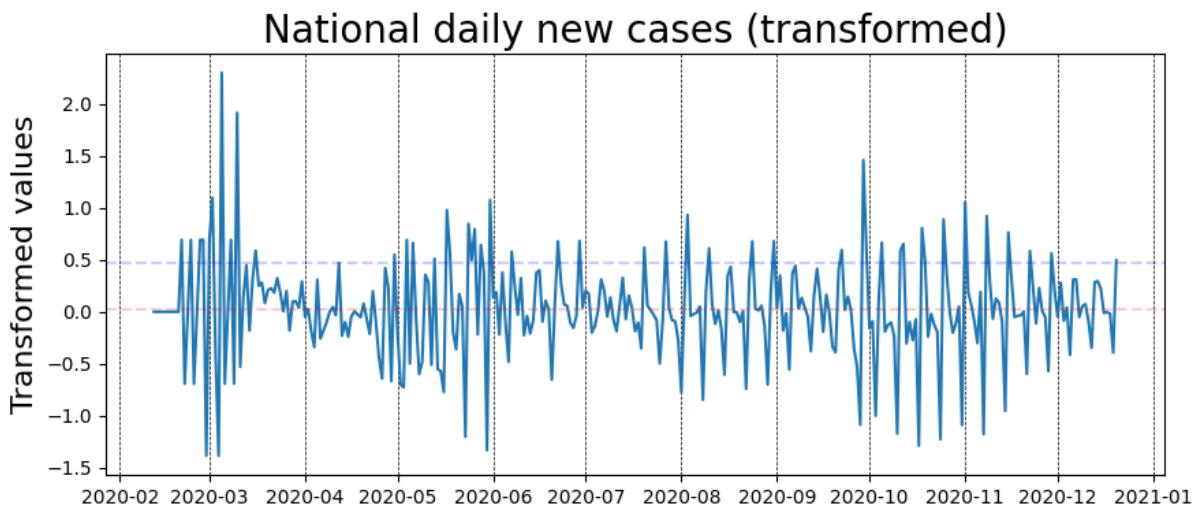
תחילה נרצה לטען את time series ולקבוע את נקודת הפיצול עבור תהליכי הלמידה. נקודה זו מסמנת את התאריך עד אליו אנחנו מאמנים את המודל שלנו, וממנו נוכל לבצע חיזוי. לצורך ההדגמה נקבע את תאריך סוף הלמידה שלנו להיות ה-20/12/2020. נתבונן על גרף הלמידה המציג את מספר המאומתים כתלות בזמן עבור תקופת התאריכים שנבחרה :



נ裏ץ את ה- ADF על מנת לבדוק האם time series שלנו סטציונירי, ונגלה (כפי שניתן לראות גם מבחן ויזואלית) כי הוא אינו סטציונירי:

```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -2.839954
P-Value                  0.052789
# Lags Used              13.000000
# Observations Used     299.000000
Critical Value (1%)      -3.452411
Critical Value (5%)       -2.871255
Critical Value (10%)      -2.571947
dtype: float64
Is the time series stationary? False
```

על מנת להפוך את time series שלנו לסטציונירי, נפעיל שתי טרנספורמציות על ה-data – קודם נפעיל את הטרנספורמציה \ln , ולאחר מכן את הטרנספורמציה $(1-diff)^{-1}$. בחרנו בטרנספורמציה אלו מכיוון שהן גם עוזרות להקטין את הסקאללהعلاיה אנחנו עובדים, ובנוסף מביאות אותנו למצב סטציונירי. נציג את ה-timeseries שלנו לאחר הפעלת הטרנספורמציות:



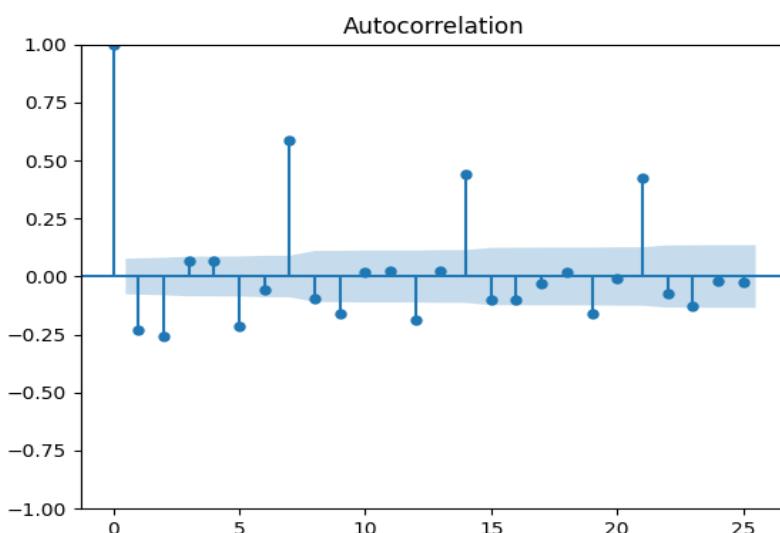
```

Is the time series stationary? False
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -4.390343
P-Value                  0.000309
# Lags Used             20.000000
# Observations Used     629.000000
Critical Value (1%)      -3.440789
Critical Value (5%)      -2.866146
Critical Value (10%)     -2.569223
dtype: float64
Is the time series stationary? True

```

הפעלת ADF test תאשר לנו שה-timeseries שלנו סטציונירי:

כעת ניתן להתחיל בתحلיך החיזוי. נפעיל את פונקציות ה-ACF על ה-timeseries שלנו ונציג את התוצאות עבור מס' lags השווה ל-25:



נזכיר כי גראף ה-ACF מציג לנו את הקורלציות עבור ה-lags השונים ב-time series שלנו. עבור $lag = 0$ תמיד נקבל כי ה-ACF שווה ל-1 מכיוון שהקורסילציה של עצה עם עצמה היא 1. כזכור, כל הערכאים שנמצאים בתחום הרצועה הכהולה נחשבים ל zenithים מבחינה סטטיסטית. מניתו התוצאות שקיבלו ניתן להסיק מס' :

- עבור $lag = 1,2,5$ נקבל אוטו-קורסילציה גבוהה. עובדה זו לא מפתיעה משום שישנה נטייה ל-lags סמוכים להיות בעלי קורלציה סטטיסטית גבוהה.
- עבור $lag = 3,4,6$ נקבל אוטו קורלציה נמוכה.
- עבור $lag = 7,14,21$ נקבל אוטו-קורסילציה גבוהה במיוחד. זהי עובדה מעניינת שמצויה על עונתיות מסוימת שקיים לנו ב-time series. ניתן לכנות עונתיות זו בתור עונתיות לוקלית. ככלומר, למרות שבמבחן time series שלנו לא מראה מגמות עונתיות ברורות, ניתן לראות כי קיים קשר בין ערכים המרוחקים זה מזה בדיקות 7 ימים. עובדה זו מתקשתה להשערות שהעלינו בפרקם הקודמים באשר להשפעה של היום בשבוע על מספר המאומתים. בפרט, הקשר שקיים בין ימי סוף השבוע בהם מבוצעות פחות בדיקות על מספר המאומתים היומי.

מניתו התוצאות נוכל להציג מס' אפשרויות להגדלת מודל ה-MA שלנו. לדוגמה:
 $(7) MA(1), MA(2), MA(7)$ או $MA(1, 1, 1)$ הן כולן אפשרויות טובות על סמך ניתוח גראף ה-ACF שקיבלו. אם נבחר ב- $(7) MA(1, 1, 1)$ אנחנו מקבלים מצד אחד התחשבות ביוטר lag-ים דומיננטיים, אך מצד שני גם התחשבות bi-lag-ים זניחים אשר עלולים להוסיף רעש לחיזוי. יתרון נוסף בהערכת lag גדול יותר היא יכולת שלנו לחזות יותר ימים בעזרת המודל המתkeletal.

נציג בעת כיצד נראים המודלים $MA(2)$ ו- $MA(1, 1, 1)$ עבור אימונו המודל. פונקציה זו מאפשרת לנו אמן מס' סוגים של אלגוריתמי time series (המאוגדים תחת השם ARIMA). במקרה זה נשתמש רק בחלק המתאים למודל ה-moving average, ובהמשך הפרק נסקר את שאר המודלים הנכללים תחת שם כולל זה.

נರיץ את מודל האימון עבור $MA(2)$. פلت האימון של המודל :

```
Model Fitting Time: 0.12680482864379883
SARIMAX Results
=====
Dep. Variable: daily_new_cases No. Observations: 313
Model: ARIMA(0, 0, 2) Log Likelihood: -188.833
Date: Sat, 11 Dec 2021 AIC: 385.665
Time: 18:50:12 BIC: 400.650
Sample: 02-12-2020 HQIC: 391.653
          - 12-20-2020
Covariance Type: opg
=====
```

	coef	std err	z	P> z	[0.025	0.975]
const	0.0255	0.014	1.806	0.071	-0.002	0.053
ma.L1	-0.3026	0.050	-6.053	0.000	-0.401	-0.205
ma.L2	-0.1749	0.051	-3.425	0.001	-0.275	-0.075
sigma2	0.1955	0.011	17.784	0.000	0.174	0.217

```

Ljung-Box (L1) (Q): 0.12 Jarque-Bera (JB): 100.06
Prob(Q): 0.72 Prob(JB): 0.00
Heteroskedasticity (H): 0.74 Skew: -0.31
Prob(H) (two-sided): 0.12 Kurtosis: 5.70
=====
```

פלט זה מציג לנו פרטים על המודל המאומן, לרבות המקדים של משווהת החיזוי וסטטיסטיות נוספות המשמשות להערכת טיב המודל. נתמקד במטריקות החשובות לנו להבנת המודל המאומן וקבלת משווהת החיזוי. ב痼ע אדום ניתן לראות את הפרטים על המודל אותו אנחנו אמנים:

- המשנה אותו אנחנו חוזים (daily_new_cases)
- סוג המודל – ARIMA(0,0,2) מסמל שאנו משתמשים במודל ARIMA אך מאנשים את שני הארגומנטים הראשונים שלו (אשר משמשים להגדרת מודלים אחרים עליהם נרחב בחלקים הבאים) ומשתמשים רק בארגומנט השלישי אשר מייצג את מודל ה-moving average עם פרמטר $q = 2$.
- Sample – מייצג את תקופת הלמידה. במקרה שלנו תקופת התאריכים עליה אנחנו מבצעים את האימון היא מה-20/02/2020 ועד ה-20/12/2020.

ב痼ע יロー ניתן לראות את הפרטים על המודל המאומן שקיבלו:

- הערך const מייצג את החיזוי של תוחלת הנתונים. זהו המקדם החופשי של המודל.
- המקדים של ערכי השגיאה מיוצגים על ידי הערכאים L1.ma ו-L2.ma.
- ערכי-P מייצגים את החשיבות הסטטיסטית של כל אחד מהמקדים של המודל. לפי כלל האכבע שהוצע בפרק בו תיארנו את hypothesis testing statistical, אם $\alpha \leq 0.05$ נאמר כי למקדם ישנה חשיבות סטטיסטית עבור החיזוי, ואם $\alpha > 0.05$ נאמר כי המקדם זניח מבחינה סטטיסטית עבור תהליך החיזוי. עבור המודל שלנו ניתן לראות כי המקדים עבור שני ה-lag-ים של המודל הם בעלי חשיבות סטטיסטית שאינה זניחה.

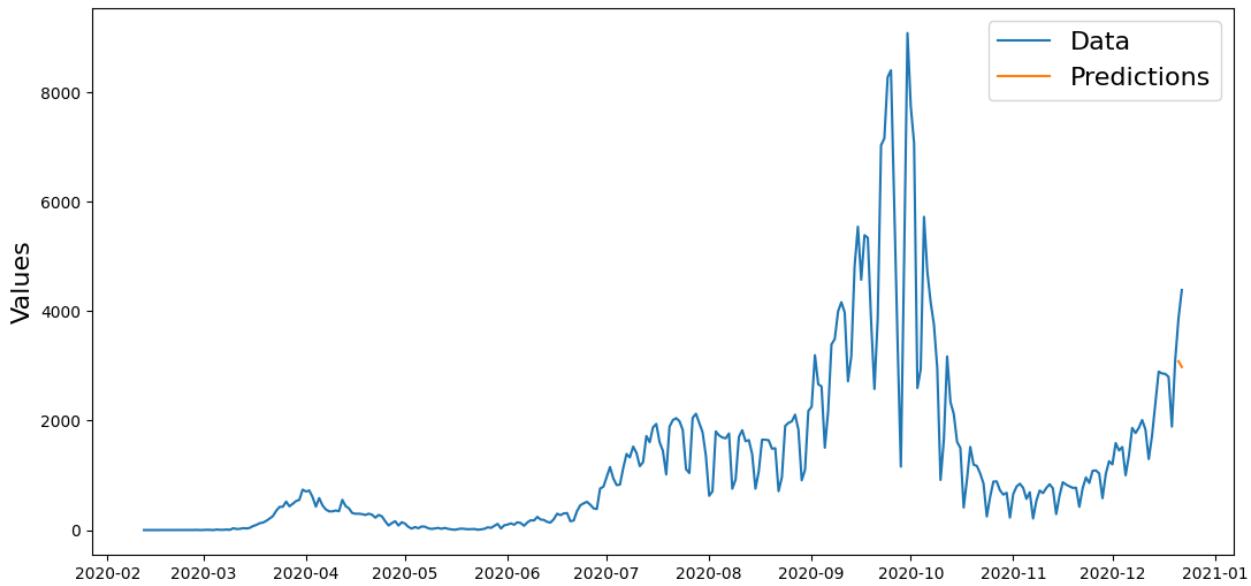
מהפלט של אימון המודל נוכל להפיק את משווהת החיזוי:

$$\hat{y}_t = 0.0255 - 0.3026\epsilon_{t-1} - 0.1749\epsilon_{t-2}$$

נשים לב כי משווהת החיזוי היא משווהה שחוצה ערכים שעברו טרנספורמציה. אם נרצה לקבל תוצאות המתאימות לתוכנה אותה אנחנו רוצים לחזות, נצטרך להפעיל את הטרנספורמציה ההפוכה על ערכי החיזוי המתוקבים.

כעת ניתן לבצע חיזוי באמצעות המודל המאומן. בהתאם למוגבלות המודל, נוכל לחזות עד יומיים עתידיים. לצורך הדוגמה נבצע חיזוי של יומיים. לאחר קבלת ערכי החיזוי נפעיל עליהם את הטרנספורמציה ההפוכה כדי לקבל את תוצאות החיזוי עבור מס' המאומתים היומי. נציג את תוצאות החיזוי על גרף ביחד עם ערכי האמת עבור מס' המאומתים:

Data vs. Predictions



```
####--Metrics for model accuracy---###
Test Data Mean: 4126.5
Mean Absolute Error 1094.4057302844526
Mean Squared Error: 1294924.8195329807
Mean Absolute Percentage Error: 0.2614969139261745
Root Mean Squared Error: 1137.9476347938778
Median Absolute Error: 1094.4057302844526
R2 score: -18.229581409825254
```

נציג את המטריקות השונות עבור דיקט המודל:

ניתן לראות כי אחזו השגיאה עומד על כ-0.26 וכי תוחלת השגיאה עומדת על כ-4.4 בהשוואה לתוחלת ערכי ה-*test set* העומדת על כ-5.4126.5.

```
Model Fitting Time: 0.6419403553009033
SARIMAX Results
=====
Dep. Variable: daily_new_cases No. Observations: 313
Model: ARIMA(0, 0, 7) Log Likelihood: -128.378
Date: Sat, 18 Dec 2021 AIC: 274.756
Time: 21:28:07 BIC: 308.472
Sample: 02-12-2020 HQIC: 288.230
- 12-20-2020
Covariance Type: opg
=====
            coef    std err      z   P>|z|   [0.025    0.975]
-----
const    0.0260    0.022    1.160    0.246   -0.018    0.070
ma.L1   -0.3654    0.046   -7.892    0.000   -0.456   -0.275
ma.L2   -0.1690    0.042   -3.999    0.000   -0.252   -0.086
ma.L3    0.1535    0.050    3.073    0.002    0.056    0.251
ma.L4   -0.1687    0.046   -3.693    0.000   -0.258   -0.079
ma.L5    0.0014    0.041    0.033    0.973   -0.078    0.081
ma.L6    0.0277    0.048    0.578    0.563   -0.066    0.121
ma.L7    0.5035    0.045   11.224    0.000    0.416    0.591
sigma2   0.1314    0.008   16.094    0.000    0.115    0.147
-----
Ljung-Box (L1) (Q):        1.38 Jarque-Bera (JB):       69.41
Prob(Q):                 0.24 Prob(JB):                  0.00
Heteroskedasticity (H):   0.66 Skew:                      0.04
Prob(H) (two-sided):      0.04 Kurtosis:                 5.31
=====
```

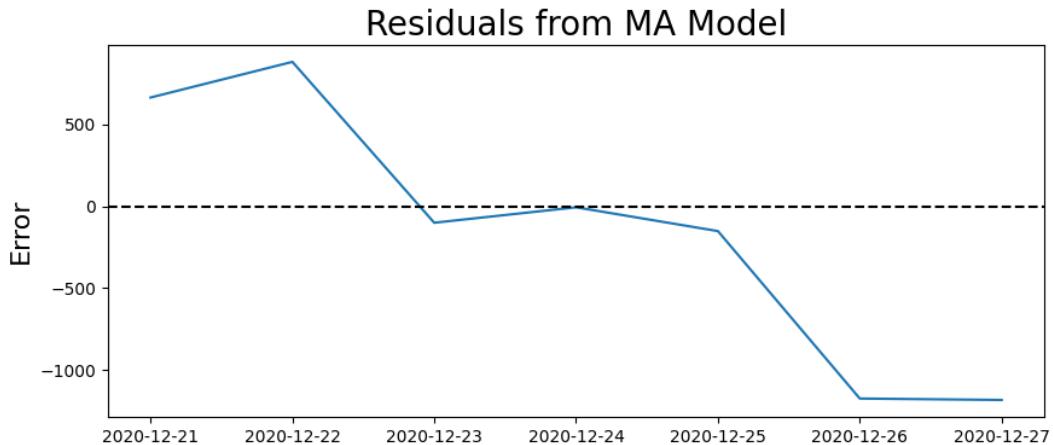
כעת נרים את מודל האימון עבור $MA(7)$. פلت מודל האימון :

ראשית, נבחן כי זמן האימון של המודל גדול כל שהערך של q גדול. בנוסף, אם נבחן את ערכי d המתקבלים עבור משווהות החיזוי ניתן לראות כי המקדים של השגיאות עבור $lags$ 5 ו-6 זניחים מבחינה סטטיסטית. עובדה זו מティישבת יפה עם ניתוח גראף ה-*ACF* שבו ניתן לראות כי עבור $lags$ 5 ו-6 אכן מתקבלים ערכי אותו קורלציה נמוכים באופן יחסי. מהפלט של אימון המודל נוכל להפיק את משווהות החיזוי :

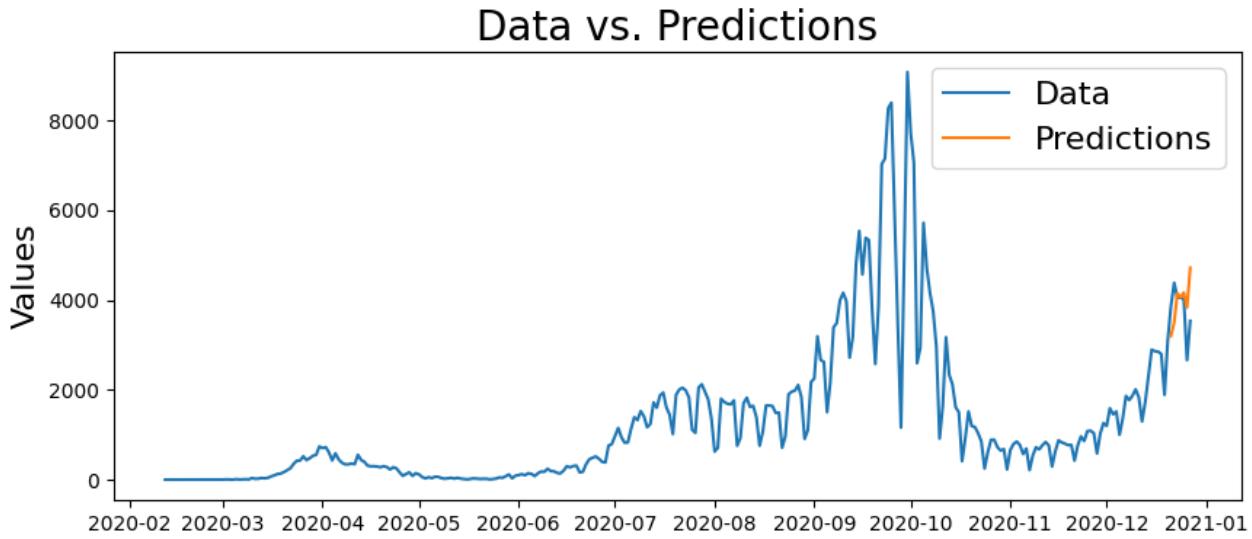
$$\hat{y}_t = 0.0260 - 0.3654\varepsilon_{t-1} - 0.1690\varepsilon_{t-2} + 0.1535\varepsilon_{t-3} - 0.1687\varepsilon_{t-4} + 0.0014\varepsilon_{t-5} \\ + 0.277\varepsilon_{t-6} + 0.5035\varepsilon_{t-7}$$

כעת ניתן לבצע חיזוי באמצעות המודל המאומן. בהתאם למוגבלות המודל, נוכל לחזות עד שבעה ימים עתידיים. לצורך הדוגמה נבצע חיזוי של שבעה ימים. לאחר קבלת ערכי החיזוי נפעיל עליהם את הטרנספורמציה הפוכה כדי לקבל את תוצאות החיזוי עבור מס' המאומתים היומי. נציג את גרפ' ה-*residuals* עבור הערכים שחווינו. גרפ' זה מייצג את השגיאה בחיזוי של היום t . עבור כל יום אותו חוותנו

$$\text{נציג את השגיאה המוחושבת ע"י: } \varepsilon_t = y_t - \hat{y}_t.$$



ניתן לראות כי אין חתנותות לנארית לגרף השגיאות. ישנו ימים בהם השגיאה מאד קטנה ומונך ימים בהם השגיאה " קופצת" בזרחה משמעותית. אחת ההשערות שלנו בנוגע להופעת השגיאות הגדולות בחיזוי היא העובדה שאנו לא מטפלים במרכיב העונתי שקיים לנו ב-time series כפי שניתחנו לעיל. בהמשך הפרק נראה אלגוריתמים אשר יידעו להתמודד יותר טוב עם סוגיה זו. כעת נציג את תוצאות החיזוי על גרפ' ביחד עם ערכי האמת עבור מס' המאומתים:



```
#####Metrics for model accuracy#####
Test Data Mean: 4090.750000000001
Mean Absolute Error 565.5409652659454
Mean Squared Error: 363452.0175737561
Mean Absolute Percentage Error: 0.13705530201554592
Root Mean Squared Error: 602.8698180981995
Median Absolute Error: 586.0799242157282
R2 score: -9.39914356893674
```

נציג את המטריקות השונות עבור דיקוק המודל :

ניתן לראות שיפור משמעותי בדיקוק הממוצע של המודל שלנו בחיזויו לשבעה ימים עם $MA(7)$.
אחוֹ השגיאה עומדת על כ-0.13 בהשוואה ל-0.26 שקיבלו עבור $MA(2)$. בנוסף תוחלת השגיאה עומדת על כ-565.5 בהשוואה לתוחלת ערכיה ה-test set העומדת על כ-4090.75.

סיכום

ב חלק זה ביצענו את "טבילת האש" הראשונה אל תוך עולם אלגוריתמי הלמידה עבור בעיות time series. הגדרנו את הבעיה אותה אנחנו מנסים לפתור והציגנו מס' נק' חשובות בניתוח ה-data, הגדרת מודל הלמידה וניתוח תוצאות האימון של המודל. חשוב לציין שאלגוריתם moving average הינו אלגוריתם בסיסי מאד, וכי יעילותו תלויות ממד בהתחנוגות של ה-data אותו רוצים לחזות. כפי שראינו כאן, התמודדות של moving average עם סטציוני עם מאפיינים עונתיים לא הייתה אופטימלית. במקרים הבאים נציג אלגוריתמים המותאמים יותר עבור הבעיה שלנו ונראה כיצד נוכל לעזורם להשיג תוצאות חיזוי טובות יותר.

Auto Regression Model

מודל ה-time series auto regression (בקיצור – AR) הוא מודל נפוץ מאד ושימושי לחיזוי בעיות מסווג משתנה הפלט במודל (הערך אותו אנחנו רוצים לחזות) תלוי לינארית בערכים קודמים של אותו המשתנה. נתאר את המודל בצורה פורמלית ולאחר מכן נסביר את האינטואיציה העומדת מאחוריו.

מודל $AR(p)$ מוגדר באופן הבא :

$$y_t = \mu + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

כאשר μ היא התוחלת של סדרת הדגימות, $\phi_1, \phi_2, \dots, \phi_q$ הם פרמטרים של המודל, $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ הם ערכי הערך של משתנה הפלט שלנו עבור $t-1, \dots, t-p$ lags והוא פרמטר השגיאה או "הרעש" (white noise error). הפרמטרים ϕ_1, \dots, ϕ_q מבטאים את המשקל של כל פרדיקט, תוך התחשבות במשקל של כל שאר הפרדיקטים על המודל. הפרדיקטים במקרה זה הם ערכי הערך של התכונה.

מהגדרת המודל ניתן לראות כי ערך של מדידה בזמן t שווה לערך התוחלת של הדגימות, בתוספת קומבינציה לינארית של ערכי עבר של התכונה אותה אנחנו רוצים לחזות. האינטואיציה במקרה זה די ברורה – אנחנו מניחים כי על מנת לחזות תכונה כלשהי בזמן t , ניתן להסתכל על ערכי התכונה בזמן $t-1$ (ובזמנים סמוכים נוספים) ולהסיק מכך את החיזוי המתבקש.

משוואת החיזוי עבור (p) :

$$\hat{y}_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p}$$

ומכאן נקבל שהשגיאה נתונה ע"י : $y_t - \hat{y}_t = \varepsilon_t$, כאשר \hat{y}_t הוא ערך החיזוי ו- y_t הוא ערך האמת.

מציאת הפרמטר c עבור מודל AR

נרצה למצוא את הפרמטר c האופטימלי עבור המודל שלנו. לשם כך ניעזר בפונקציית ה-PACF (Partial Autocorrelation).

תזכורת : באופן דומה לפונקציית ACF, גם פונקציית PACF מודדת קורלציה בין מדידות שונות. ההבדל העיקרי בין ACF ל-PACF הוא שפונקציית PACF מודדת את הקורלציה הישירה בין שתי המדידות, בעוד פונקציית PACF לוקחת בחשבון גם את ההשפעה של המדידות האחרות הנמצאות בטוחה שבין x_t ל- x_{t-k} . בכך אנחנו מקבלים קורלציה אשר מתחשבת **במיוחד** של המדידות בתוך ה-time series.

פורמלית – פונקציית PACF עבור זמן t ו- k lag תוגדר באופן הבא :

$$PACF(t, k) = \frac{cov(x_t, x_{t-k} | x_{t-1}, x_{t-2}, \dots, x_{t-k+1})}{\sqrt{var(x_t | x_{t-1}, x_{t-2}, \dots, x_{t-k+1}) \cdot var(x_{t-k} | x_{t-1}, x_{t-2}, \dots, x_{t-k+1})}}$$

כפי שראינו בפרק הקודם, גם כאן קיים tradeoff בבחירה הפרמטר c עבור המודל שלנו.

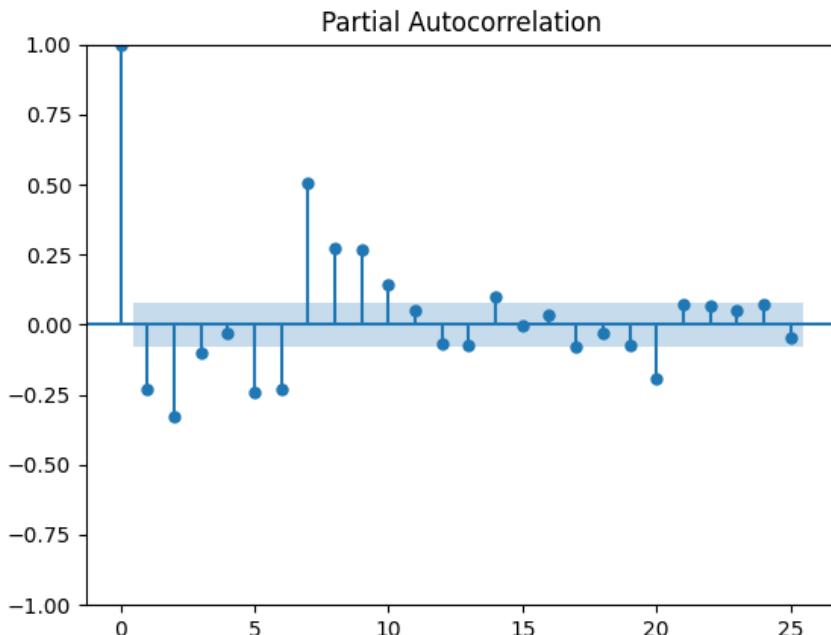
חיזוי באמצעות (p) AR

לאחר בחירת הפרמטר p עבור המודל, יבוצע תהליך למידה בו יבחרו הפרמטרים עבור משווהות החיזוי. לאחר מכן נוכל לבחור את התקופה אותה נרצה לחזות. כאשר נבצע חיזוי באמצעות (p) AR, ערכי התוכנות עבור lags "מה עבר" (כלומר ה-lags שבזורתם אנחנו חוזים את היום הנוכחי) יילקחו מה-data המקורי כל עוד ערך ה-lag הרלוונטי נמצא בתחום ה-set של המודל. במידה וערך התוכנה הקודמת נופל על lag שגム הוא נמצא בתחום החיזוי (כלומר ערך שאנו לא יודעimos את ערכו האמתי), נשתמש בערך החיזוי עבור אותו ה-lag. בכך המודל יוכל בצורה איטרטיבית לחזות את ערכי התוכנה המבוקשת.

נראה כעת דוגמה בסיסית לחיזוי באמצעות מודל regression auto, ונראה את שלביים השונים החל מטעינת ה-data, אימון המודל וניתוח התוצאות. לצורך הדוגמה השתמש שוב ב-time series שהוצג בחלק הקודם, המציג את מס' המאומטמים ברמה הלאומית. נשתמש באותו הטרנספורמציות שביבצנו על ה-data כדי להביא אותו למצב סטציוני ובאותן תקופות תאריכים עבור תהליכי האימון.

כעת ניתן להתחיל בתהליך החיזוי. נפעיל את פונקציית `pacf` על ה-time series PACF שלנו ונציג את התוצאות בגרף עבור מס' lags השווה ל-25 :

מניתוח התוצאות שקיבלנו ניתן להסיק מס' מסקנות :



- עבור $lag = 1,2,5,6,7$ קיבל אוטו-קורלציה גבוהה.
- עבור $lag = 3,4$ קיבל אוטו קורלציה נמוכה.
- כפי שראינו בחלק הקודם, גם כאן עבור $lag = 7$ קיבל אוטו-קורלציה גבוהה במיוחד.

מניתוח התוצאות נוכל להציג מס' אפשרויות להגדרת מודל ה-AR שלנו. לדוגמה : (AR(2), AR(5), AR(7)) .
הן כולן אופציונות טובות על סמך ניתוח גרפ-pacf שקיבלנו.

נראה כיצד נראה מודל $AR(2)$ ומבצע חיזוי של מס' ימים עתידיים. משתמש בפונקציה ARIMA עבור אימון המודל. פונקציה זו מאפשרת לאמן מס' סוגים של אלגוריתמי time series. בפרק זה משתמש רק בחלק המתאים למודל ה-.auto regression

נ裏 את מודל האימון עבור $AR(2)$. פلت האימון של המודל:

```
Model Fitting Time: 0.08367633819580078
SARIMAX Results
=====
Dep. Variable: daily_new_cases No. Observations: 313
Model: ARIMA(2, 0, 0) Log Likelihood: -190.044
Date: Sat, 25 Dec 2021 AIC: 388.087
Time: 19:48:44 BIC: 403.072
Sample: 02-12-2020 HQIC: 394.076
- 12-20-2020
Covariance Type: opg
=====
            coef    std err        z      P>|z|      [0.025      0.975]
-----
const    0.0254    0.017     1.455     0.146     -0.009     0.060
ar.L1   -0.2925    0.046    -6.396     0.000     -0.382    -0.203
ar.L2   -0.2700    0.048    -5.682     0.000     -0.363    -0.177
sigma2   0.1971    0.011   17.388     0.000      0.175     0.219
=====
Ljung-Box (L1) (Q): 0.09 Jarque-Bera (JB): 79.87
Prob(Q): 0.77 Prob(JB): 0.00
Heteroskedasticity (H): 0.85 Skew: -0.37
Prob(H) (two-sided): 0.41 Kurtosis: 5.36
=====
```

ננתח את פلت האימון באותו האימון כפי שראינו בחלק הקודם עבור מודל ה-.moving average. נשים לב כי השהסימון של המודל עם הפרמטרים $ARIMA(2,0,0)$ מראה שאנו מרכיבים מודל auto regression עם $2 = d$. בנוסף, מנתוח המקדמים של המודל (המסומנים עם התחלית 'L', 'ar',), ניתן לראות כי במקרה זה לכל המקדמים חשיבות סטטיסטית לא זניחה על החיזוי.

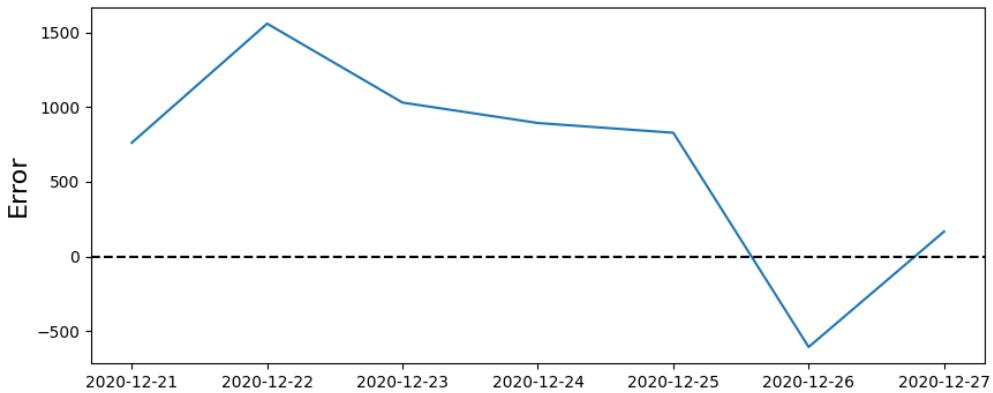
מהפלט של אימון המודל נוכל להפיק את משווהת החיזוי:

$$\hat{y}_t = 0.0254 - 0.2925y_{t-1} - 0.2700y_{t-2}$$

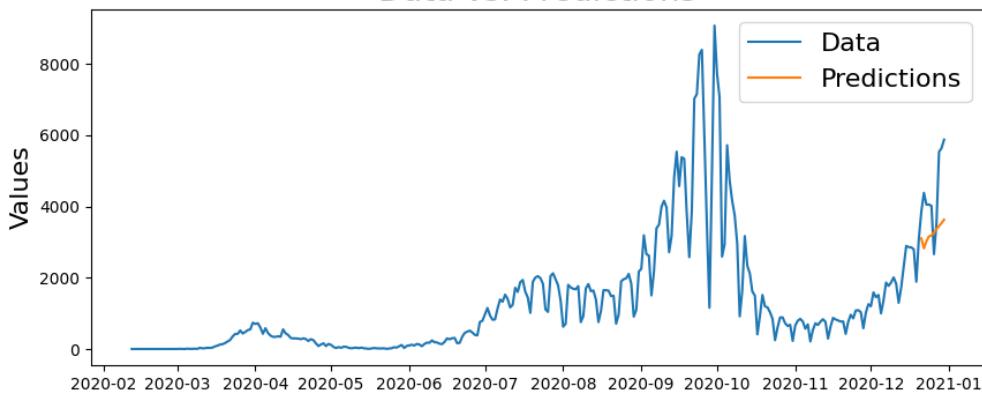
נזכיר כי משווהת החיזוי היא משווהה שהזוהה ערכיהם בעברו טרנספורמציה. כדי לקבל תוצאות המתאימות לתוכונה אותה אנחנו רוצים לחזות, נפעיל את הטרנספורמציה ההופוכה על ערכי החיזוי המתוקבלים.

כעת ניתן לבצע חיזוי באמצעות המודל המאומן. לצורך הדוגמה נזהה כתת תקופה של 7 ימים, החל מסוף תקופת האימון. נציג כפלט את גրף השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:

Residuals from AR Model



Data vs. Predictions



```
##### Metrics for model accuracy #####
Test Data Mean: 3797.285714285714
Mean Absolute Error 835.7121454066861
Mean Squared Error: 851512.1653427128
Mean Absolute Percentage Error: 0.2155575919466925
Root Mean Squared Error: 922.774168116291
Median Absolute Error: 828.9381992868266
R2 score: -2.1581711549970106
```

נציג את המטריקות השונות עבור דיקט המודל:

ניתן לראות כי אחוז השגיאה עומד על כ-0.21 וכי תוחלת השגיאה עומדת על כ-835.7 בהשוואה לתוחלת ערכי ה-test set העומדת על כ-3797.2.

כעת נרים את מודל האימון עבור (7). מהסתכלות על פلت המודל ניתן לראות שתת כי חלק מהמקדמים של המודל שלנו זניחים מבחינה סטטיסטית.

משוואת החיזוי:

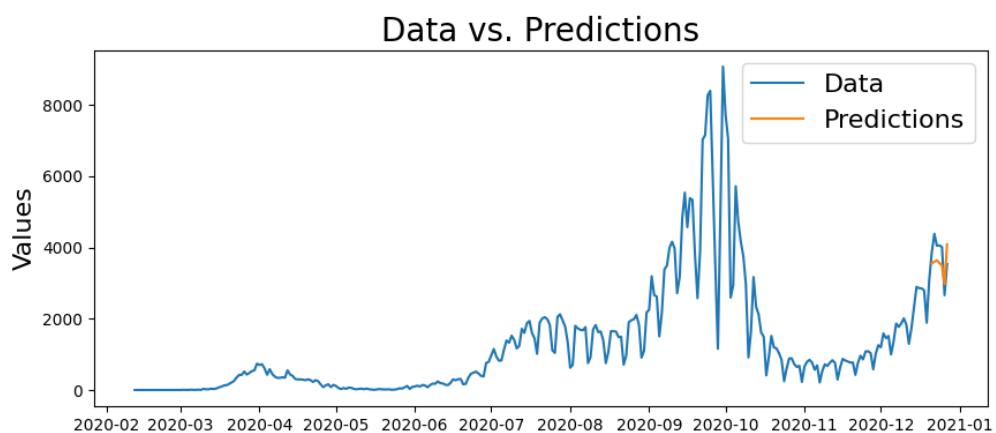
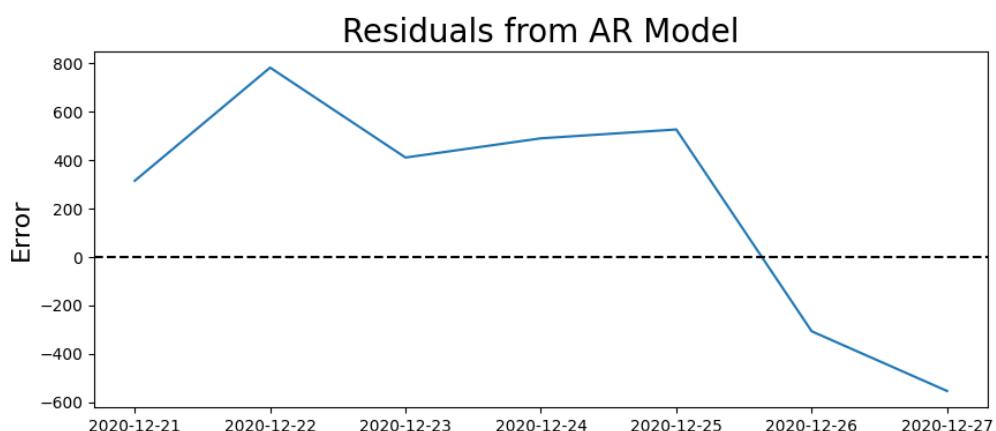
$$\hat{y}_t = 0.0257 - 0.2726y_{t-1} - 0.2388y_{t-2} - 0.0678y_{t-3} - 0.1301y_{t-4} - 0.0684y_{t-5} \\ - 0.0091y_{t-6} + 0.4834y_{t-7}$$

```

Model Fitting Time: 0.355482816696167
                    SARIMAX Results
=====
Dep. Variable: daily_new_cases    No. Observations:            313
Model:          ARIMA(7, 0, 0)    Log Likelihood:           -134.625
Date:          Sat, 25 Dec 2021   AIC:                      287.251
Time:          20:14:16          BIC:                      320.967
Sample:         02-12-2020      HQIC:                     300.724
                           - 12-20-2020
Covariance Type: opg
=====
              coef    std err        z    P>|z|    [0.025    0.975]
-----
const       0.0257    0.018     1.455    0.146    -0.009    0.060
ar.L1      -0.2726    0.040    -6.772    0.000    -0.351   -0.194
ar.L2      -0.2388    0.040    -5.970    0.000    -0.317   -0.160
ar.L3      -0.0678    0.044    -1.528    0.127    -0.155    0.019
ar.L4      -0.1301    0.051    -2.528    0.011    -0.231   -0.029
ar.L5      -0.0684    0.047    -1.465    0.143    -0.160    0.023
ar.L6      -0.0091    0.043    -0.210    0.834    -0.094    0.076
ar.L7       0.4834    0.041    11.687    0.000    0.402    0.564
sigma2      0.1373    0.007    19.715    0.000    0.124    0.151
-----
Ljung-Box (L1) (Q):      7.60    Jarque-Bera (JB):      364.73
Prob(Q):                  0.01    Prob(JB):                   0.00
Heteroskedasticity (H):  0.44    Skew:                      0.40
Prob(H) (two-sided):     0.00    Kurtosis:                 8.23
=====
```

ניתן לראות כאן ביטוי לקשר בין פונקציית ה-PACF למשוואת החיזוי – עבור ה-lags בהם פונקציית ה-PACF הניבה קורלציה שלילית, ניתן לראות כי מקדם החיזוי המתקבל הוא שלילי, ועבור lag = 7 שהוא היחיד בו פונקציית ה-PACF הניבה קורלציה חיובית, כך גם המקדם של lag זה במשוואת החיזוי הוא חיובי.

נבעו שוב חיזוי של 7 ימים ונציג את התוצאות:



```
##### Metrics for model accuracy#####
Test Data Mean: 3797.285714285714
Mean Absolute Error 483.7062595183062
Mean Squared Error: 256998.06013234027
Mean Absolute Percentage Error: 0.12641716692739077
Root Mean Squared Error: 506.9497609550085
Median Absolute Error: 490.4147114362868
R2 score: 0.04682059348679124
```

נציג את המטריקות השונות עבור דיקוק המודל :

ניתן לראות שת כ- 483.7 מילימטרים השגיאה עומדת על כ- 0.12 ומי תוחלת השגיאה עומדת על כ- 3797.2. ביחסוואה לתוחלת ערכי ה-test set העומדת על

סיכום

ב חלק זה ראיינו את מודל ה-*auto regression*. זהו מודל חיזוי קלאסי ושימושי מאד לבעיות *time series* והוא פועל בצורה מאד אינטואיטיבית המאפשרת לחזות תכונה לפי הערכים הקודמים של אותה התכונה. בנוסף ראיינו את חשיבות פונקציית ה-PACF בبنית מודל החיזוי. ב חלק הבא נעבור לאלגוריתם המשלב את שני האלגוריתמי החיזוי שהציגנו עד כה.

ARMA Model

בחלקים הקודמים הצגנו את המודלים AR ו-MA לפרטון בעיות time series. לכל אחד מהמודלים משווהת חיזוי לינארית שונה אשר מtabסת על פרדיקטים (predictors) שונים. כתה נרצה להציג פתרון אשר משלב את שני האלגוריתמים הללו.

מודל (auto regression moving average) $ARMA(p, q)$ משלב את שני האלגוריתמים הללו למודל אחד. הפרמטר p הוא הפרמטר עבור החלק של ה- y_t , והפרמטר q הוא הפרמטר עבור החלק של ה- ε_t : $ARMA(p, q)$. משווהת המודל עבור moving average

$$y_t = \mu + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varphi_1 \varepsilon_{t-1} + \cdots + \varphi_q \varepsilon_{t-q} + \varepsilon_t$$

כאשר μ היא התוחלת של סדרת הדגימות, $\phi_1, \phi_2, \dots, \phi_p$ הם הפרמטרים של המודל AR- y_t , $\varphi_1, \varphi_2, \dots, \varphi_q$ הם הפרמטרים של מודל MA- y_t , $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ הם ערכי הערך של משתנה הפלט שלנו עבור $t - lags$. (ε_t white noise error).

משווהת החיזוי עבור $ARMA(p, q)$:

$$\hat{y}_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varphi_1 \varepsilon_{t-1} + \cdots + \varphi_q \varepsilon_{t-q}$$

ומכאן נקבל שהשגיאה נתונה ע"י: $y_t - \hat{y}_t = \varepsilon_t$, כאשר \hat{y}_t הוא ערך החיזוי ו- y_t הוא ערך האמת.

מציאת הפרמטרים q, p עבור מודל ARMA

קבעת הפרמטרים עבור מודל ARMA יתבצע באופן זהה לזו שבייצעו בחלקים הקודמים. את הפרמטר p המתאים ל-auto regression נקבע באמצעות פונקציית ACF, ואת הפרמטר q המתאים ל-moving average.

חיזוי באמצעות $ARMA(p, q)$

לאחר בחירת הפרמטרים עבור המודל, יתבצע תהליך מיידי בו יבחרו הפרמטרים עבור משווהת החיזוי. לאחר מכן נוכל לבחור את התקופה אותה נרצה לחזות.

נראה כת דוגמה בסיסית לחיזוי באמצעות מודל ARMA. לצורך הדוגמה נשתמש שוב ב-time series שהוצג בתחילת הפרק, המציג את מס' המאומתים ברמה הלאומית. נשתמש באותו הטרנספורמציות שביצעו על ה-data כדי להביא אותו למצב סטציוני ובעותן תקופות תאריכים עבור תהליך האימון. מכיוון שאנו משתמשים באותו ה-data מהחלקים הקודמים, כל המסקנות שהסקנו לגבי פונקציית ACF וה-PACF תקופות גם כן. מניתוח פונקציות הקורלציה, ניתן לראות כי בחירה טובה עבור המודל שלנו תהיה: $p = 7, q = 7$.

נראה כיצד נראה מודל $ARMA(7,7)$ עבור ARIMA(7,0,7). ימים עתידיים. השתמש בפונקציה `forecast` אימון המודל. פונקציה זו מאפשרת לאמן מס' סוגים של אלגוריתמי time series. בפרק זה השתמש רק בחלק המתאים למודל ARMA.

נ裏 את מודל האימון עבור $ARMA(7,7)$. פلت האימון של המודל:

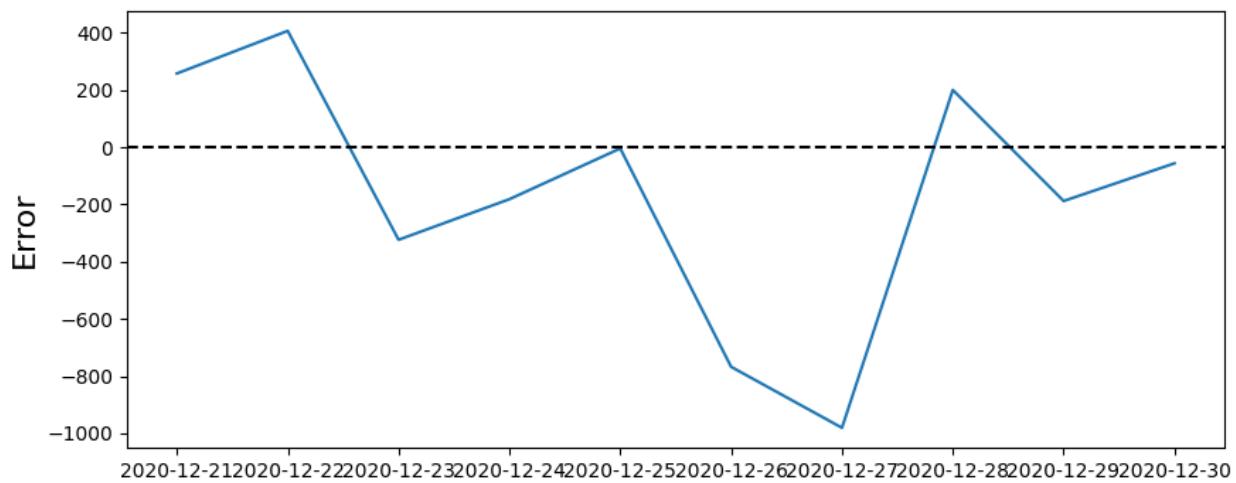
Model Fitting Time: 1.3908424377441406 SARIMAX Results						
Dep. Variable:	daily_new_cases	No. Observations:	313			
Model:	ARIMA(7, 0, 7)	Log Likelihood	-102.251			
Date:	Sun, 26 Dec 2021	AIC	236.501			
Time:	00:45:07	BIC	296.441			
Sample:	02-12-2020 - 12-20-2020	HQIC	260.455			
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
const	0.0266	0.033	0.811	0.417	-0.038	0.091
ar.L1	0.2249	0.127	1.766	0.077	-0.025	0.475
ar.L2	-0.1232	0.154	-0.800	0.424	-0.425	0.179
ar.L3	-0.0719	0.117	-0.616	0.538	-0.301	0.157
ar.L4	0.0818	0.103	0.792	0.428	-0.121	0.284
ar.L5	-0.1437	0.093	-1.545	0.122	-0.326	0.039
ar.L6	0.2292	0.089	2.587	0.010	0.056	0.403
ar.L7	0.4299	0.086	5.000	0.000	0.261	0.598
ma.L1	-0.7532	0.130	-5.807	0.000	-1.007	-0.499
ma.L2	0.0781	0.211	0.369	0.712	-0.336	0.493
ma.L3	0.2516	0.171	1.470	0.142	-0.084	0.587
ma.L4	-0.3171	0.139	-2.281	0.023	-0.590	-0.045
ma.L5	0.2840	0.150	1.889	0.059	-0.011	0.579
ma.L6	-0.2049	0.175	-1.172	0.241	-0.548	0.138
ma.L7	0.2067	0.126	1.637	0.102	-0.041	0.454
sigma2	0.1109	0.006	18.169	0.000	0.099	0.123

מניתו פلت אימון המודל ניתן לראות את ההבחנה בין המקדים של החלק השיק' ל-moving average (עם תחילית 'L.ma') לבין המקדים של החלק השיק' ל-auto regression (עם תחילית 'L.ar'). כמו כן ניתן לראות כי גם כאן חלק מהמקדים זניחים מבחינה סטטיסטית.

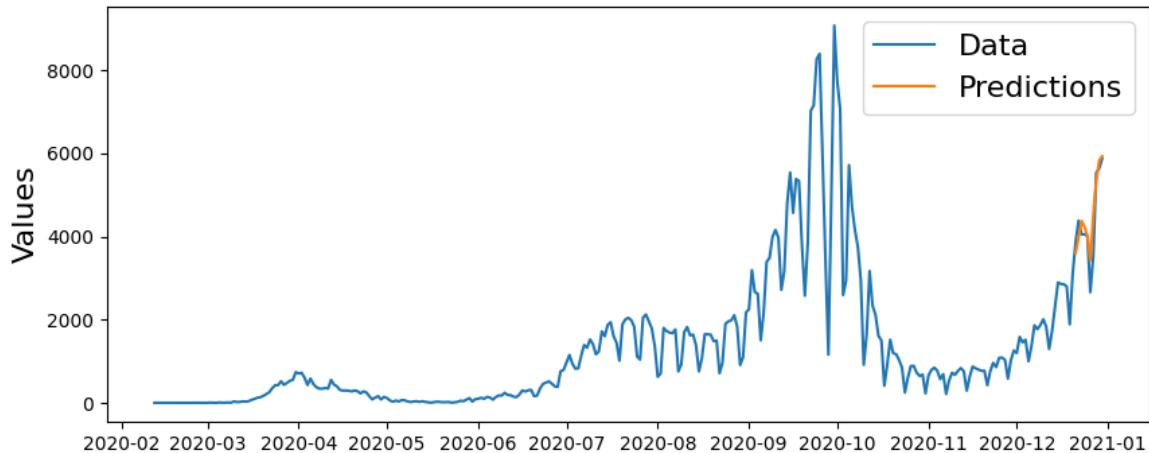
באופן לא מפתיע ניתן לראות גם כי זמן האימון של המודל גדול בהשוואה לחלקים הקודמים בהם אנחנו מאמנים רק חלק אחד (AR או MA). מפלט האימון יוכל להפיק את משווהת החיזוי שלנו (לא נרשות אותה במפורש מפאת גודלה וריבוי המשתנים במשווהה, אך הראיינו כיצד לעשות זאת בחלקים הקודמים).

כעת ניתן לבצע חיזוי באמצעות המודל המאומן. לצורך הדוגמה נחזה כתת תקופה של 10 ימים, החל מסוף תקופת האימון. נציג כפלט את גרף השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:

Residuals from ARMA Model



Data vs. Predictions



```
#####
# Metrics for model accuracy
#####
Test Data Mean: 4363.799999999999
Mean Absolute Error 336.6643976942876
Mean Squared Error: 199912.7730680147
Mean Absolute Percentage Error: 0.09296155201566515
Root Mean Squared Error: 447.11606218968996
Median Absolute Error: 228.91647191169727
R2 score: 0.7881789796062394
```

מציג את המטריקות השונות עבור דיקט המודל :

ניתן לראות כי למראות שביצענו חיזוי של 10 ימים (בממוצע לחיזוי של 7 ימים שביצענו בחלקים הקודמים), קיבלנו תוצאות טובות יותר בהשוויה להערכת המודלים AR ו- MA בנפרד על אותן תקופות התאריכים החל מסוף תקופת המבחן.

אחוז השגיאה עומד כעת על כ- 0.09 , תוחלת השגיאה עומדת על כ- 336.7 וחציון השגיאה עומד על 9,228.9 כל זאת בהשוואה לתחלת ערכי ה-test set העומדת על כ- 4363.8. גם לפי מטריקת *R2 score* קיבלנו כעת דיקט של 0.788

סיכום

בחלק זה רأינו כיצד ניתן לשלב את שני האלגוריתמי החזויים שראינו עד כה בפרק זה, וראינו כי ההילוב של שני האלגוריתמים מצליח לתת חיזוי טוב יותר בהשוואה לחיזוי של כל אחד מהמודלים בנפרד. בהמשך הפרק נמשיך לעבד על הבסיס של האלגוריתמים שהוצעו עד כה ולהציג שיפורים ואופטימיזציות נוספות שיעזרו לנו לשפר את דיוק החיזוי.

ARIMA Model

מודל ARIMA (auto regression integrated moving average) ARIMA שהציגו בחלק הקודם, בתוספת היכולת של המודל להתחאים את עצמו ל-time series שאינו סטציוני. בוגוד מודלים שהציגו עד כה, אשר מניחים כי ה-data המזון אליום כבר במצב סטציוני, מודל-h ARIMA לא דורש זאת. על מנת להביא את ה-data במצב סטציוני, המודל משתמש בטרנספורמציה (d) כפיה שהוגדרה בפרק הטרנספורמציות.

מודל ARIMA(p, d, q) מכיל שלושה פרמטרים. הפרמטר d הוא הפרמטר עבור החלק של ה-auto regression, הפרמטר d הוא הפרמטר עבור טרנספורמציה ($diff$), והפרמטר q הוא הפרמטר עבור החלק של ה-moving average. משווה את המודל עבור ARIMA(p, d, q) :

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varphi_1 \varepsilon_{t-1} + \cdots + \varphi_q \varepsilon_{t-q} + \varepsilon_t$$

כאשר ϕ_1, \dots, ϕ_p הם הפרמטרים של המודל ה-AR, $\varphi_1, \dots, \varphi_q$ הם הפרמטרים של מודל ה-MA, $y_{t-1}, y_{t-2}, \dots, y_{t-p}$ הם ערכי העבר של משתנה הפלט שלנו עבור lags $t-1, \dots, t-p$ ו- $\varepsilon_t, \varepsilon_{t-1}, \dots, \varepsilon_{t-q}$ הם פרמטרי השגיאה או "הרעש" (white noise error).

משווה את החיזוי עבור ARIMA(p, d, q) :

$$\hat{y}_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \varphi_1 \varepsilon_{t-1} + \cdots + \varphi_q \varepsilon_{t-q}$$

ומכאן נקבל שהשגיאה נתונה ע"י: $y_t - \hat{y}_t = \varepsilon_t$, כאשר \hat{y}_t הוא ערך החיזוי ו- y_t הוא ערך האמת. נשים לב שבביגוד למודל ARMA, כתע משווה את המודל ומשווה את החיזוי לא מכילה מקדם חופשי. זאת מכיוון שהמודל מכוון לכך שהפעלת הטרנספורמציה תביא את ה-data במצב בו תוכלת סדרת הדגימות שווה ל-0, וכן לא מוסיפים מקדם חופשי למשווה את החיזוי.

מציאת הפרמטרים ARIMA(p, d, q) עבור מודל

מודל ARIMA מורכב ממודל ARMA בתוספת טרנספורמציה המופעלת על ה-data. הפרמטרים p, q יקבעו באופן זהה לאופנו בו קבענו את הפרמטרים עבור מודל ARMA. עבור הפרמטר d , נקבע אותו בהתאם לאופי ה-time series שלנו. לרוב על מנת להביא את ה-time series במצב סטציוני נחוג לקבוע $d = 1$. דוגמה לכך ניתן לראות בפרק הטרנספורמציות, שם הדגמנו כיצד הפעלת הטרנספורמציה ($diff(1)$ הפכה את ה-time series הלא סטציוני שלנו לסטציוני).

חיזוי באמצעות ARIMA(p, d, q)

לאחר בחירת הפרמטרים עבור המודל, יבוצע תהליך מיידה בו יבחרו הפרמטרים עבור משווה את החיזוי. לאחר מכן נוכל לבחור את התקופה אותה נרצה לחזות. עבור המודלים הקודמים שהציגו בפרק זה, ראיינו כי علينا להפעיל טרנספורמציות על ה-data שלנו טרם הזנתו למודל. המשמעות היא שערכי החיזוי שנקבעו הם ערכיים שעברו טרנספורמציה, ולכן נאלצנו לבצע שלב נוסף בו אנחנו מפעילים את הטרנספורמציה

ההפוכה על ערכי החיזוי כדי לקבל מספרים אשר תואימים את התוכונה אותה אנחנו מנסים לנבא. בעת השימוש במודל ARIMA נחsettת לנו העבודה הכרוכה בהפעלת הטרנספורמציות והיפוכן, וכן נוכל לטעון את time series הגולמי בתוך המודל ולבצע את החיזוי ישירות.

נראה כעת דוגמה בסיסית לחיזוי באמצעות מודל ARIMA. לצורך הדוגמה נשתמש שוב ב-time series שהוצג בתחילת הפרק, המציג את מס' המאומתים ברמה הלאומית. כדי לקבוע את הפרמטר d , נפעיל את הטרנספורמציה ה- d -time series שלנו ונבחן את מידת הסטציאונריות. שלב זה הוא לא חובה כמובן, אך כדי לבצע בחירה מושכלת של הפרמטר d נרצה להראות את השפעת הטרנספורמציה על ה-data. לצורך הדוגמה נפעיל את הטרנספורמציה ה- d -time series עבור $d \in \{1, 2\}$, ונשתמש ב-ADF test כדי לבדוק את מידת הסטציאונריות של ה-time series המתkeletal.

ניתן לראות מתוצאות ה-ADF test כי שתי הטרנספורמציות מביאות את ה-time series שלנו למצב סטציאוני, אך עבור $d = 2$ קיבל ערך P-value נמוך יותר, לכן נבחר פרמטר זה עבור מודל ה-ARIMA שלנו.

עבור בחירת הפרמטרים d , נסתפק בבחירה של אותם הפרמטרים אitem הרצינו את מודל ה-ARMA. הינו יכולים לבצע התאמת מחודשת של פרמטרים אלו לפי פונקציות ה-ACF וה-PACF, אך לא נתעכבר על כך בחלק זה מכיוון שמטרתנו היא להראות את המאפיינים הייחודיים למודל ה-ARIMA.

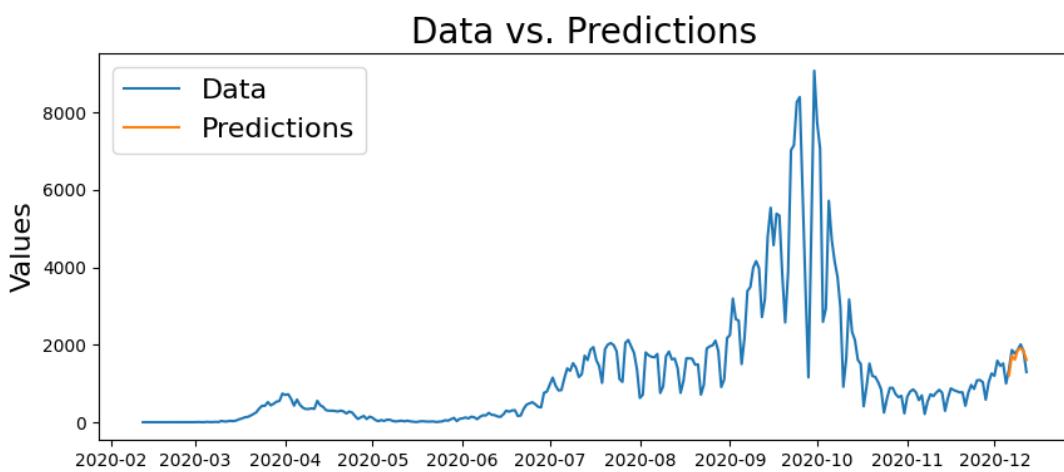
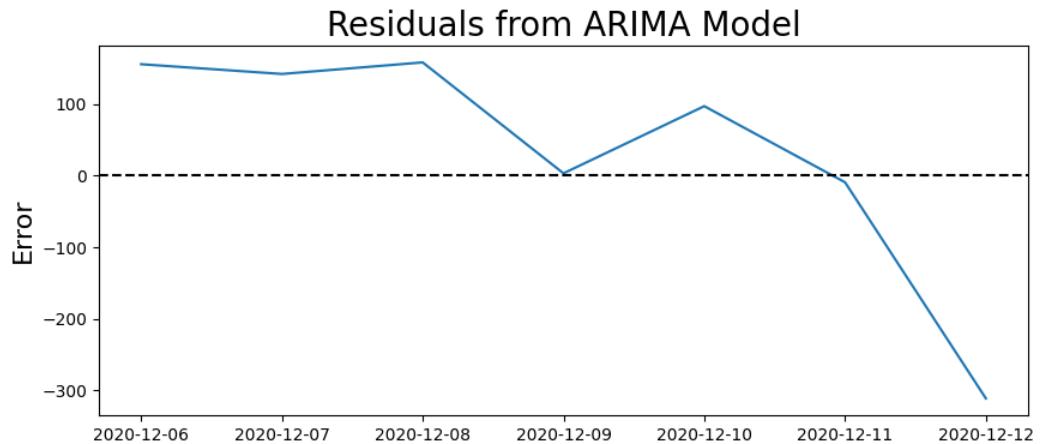
נראה כיצד נראה מודל ARIMA(7,2,7) וביצוע חיזוי של מס' ימים עתידיים. נשתמש בפונקציה ARIMA עבור אימון המודל.

נريץ את מודל האימון עבור ARIMA(7,2,7). פلت האימון של המודל:

Model Fitting Time: 1.493384838104248						
SARIMAX Results						
Dep. Variable:	daily_new_cases	No. Observations:	298			
Model:	ARIMA(7, 2, 7)	Log Likelihood	-2246.841			
Date:	Tue, 28 Dec 2021	AIC	4523.681			
Time:	00:49:01	BIC	4579.037			
Sample:	02-12-2020	HQIC	4545.844			
	- 12-05-2020					
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
ar.L1	-0.7630	0.242	-3.156	0.002	-1.237	-0.289
ar.L2	-0.7648	0.210	-3.649	0.000	-1.176	-0.354
ar.L3	-0.4900	0.266	-1.842	0.066	-1.011	0.031
ar.L4	-0.3840	0.186	-2.066	0.039	-0.748	-0.020
ar.L5	-0.4660	0.179	-2.601	0.009	-0.817	-0.115
ar.L6	-0.4927	0.158	-3.113	0.002	-0.803	-0.182
ar.L7	0.1365	0.135	1.013	0.311	-0.128	0.401
ma.L1	-0.3125	0.235	-1.328	0.184	-0.774	0.149
ma.L2	-0.5933	0.088	-6.642	0.000	-0.755	-0.411
ma.L3	-0.3159	0.120	-2.629	0.009	-0.551	-0.080
ma.L4	-0.0805	0.130	-0.618	0.537	-0.336	0.175
ma.L5	0.7354	0.095	7.774	0.000	0.550	0.921
ma.L6	0.2931	0.147	1.993	0.046	0.005	0.581
ma.L7	-0.4009	0.083	-4.836	0.000	-0.563	-0.238
sigma2	2.524e+05	1.28e+04	19.734	0.000	2.27e+05	2.77e+05
Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2850.57			
Prob(Q):	0.98	Prob(JB):	0.00			
Heteroskedasticity (H):	95.20	Skew:	-2.14			
Prob(H) (two-sided):	0.00	Kurtosis:	17.59			

ניתן לראות כי רוב הפרמטרים של המודל אינם זניחים סטטיסטית, דבר המרמז על בחירה טובה של פרמטרים עבור המודל שלנו. בנוסף ניתן לראות כי פלט המודל אינו כולל מקדם חופשי במשוואת החיזוי. מפלט האימון נוכל להפיק את משוואת החיזוי שלנו (לא רשום אותה במפורש מפאת גודלה וריבוי המשתנים במשוואה, אך הראיינו כיצד לעשות זאת בחלוקת הקודמים).

כעת ניתן לבצע חיזוי באמצעות המודל המאומן. לצורך הדוגמה נחזה בעת תקופה של 7 ימים, החל מסוף תקופת האימון. משיקולי נוחות בדוגמה זו תקופת האימון מסתיימת בתאריך 2020/05/12/2020, ובהתאם תקופת המבחן מסתיימת בתאריך 2020/05/12. נציג כפלט את גרפ' השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת :



נציג את המטריקות השונות עבור דיקוק המודל :

```
####--Metrics for model accuracy---###
Test Data Mean: 1716.7142857142858
Mean Absolute Error 125.41967809077899
Mean Squared Error: 25168.38416738043
Mean Absolute Percentage Error: 0.08214257083088293
Root Mean Squared Error: 158.64546689830263
Median Absolute Error: 142.31411739102532
R2 score: 0.6070312054449869
```

אחוז השגיאה עומד בערך 0.08 , תוחלת השגיאה עומדת בערך 125.4 בהשוואה לתוחלת ערכי ה-test set .1716.7

סיכום

מודל ARIMA מאגד בתוכו את כל האלמנטים שראינו עד כה במהלך הפרק, ומאפשר לבצע את תהליכי האימון והחיזוי בצורה נוחה וקלה, מבלתי שנייאלי' לבצע עיבודים נוספים נתונים שלנו לפני טיעינתם למודל ובעת תרגום תוצאות החיזוי.

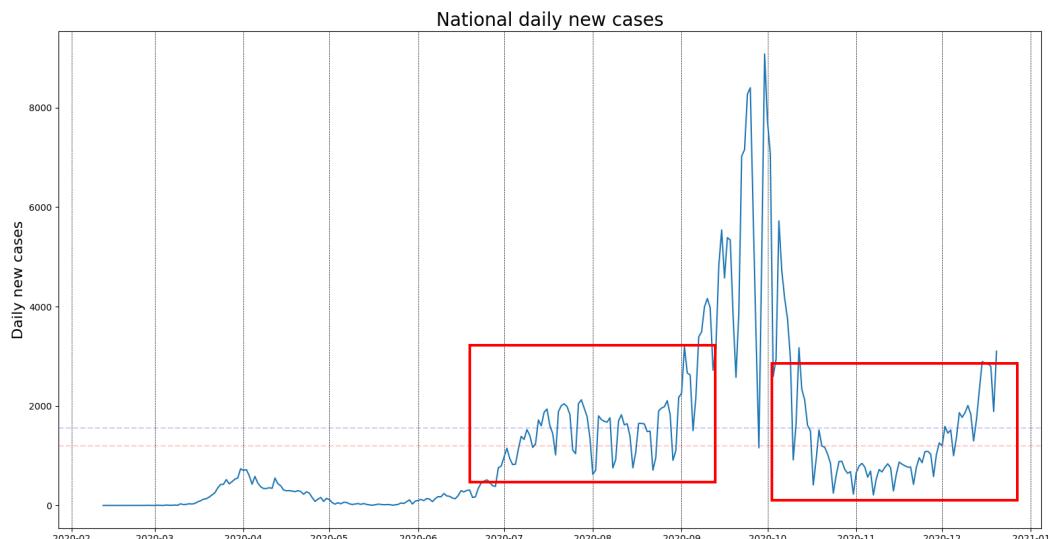
SARIMA Model

אחת הביעות הנפוצות ביותר בחיזוי בעיות time series היא בעיית העונתיות (seasonality).

תזכורת:

עונתיות (Seasonality) – מושג זה מתיחס לדפוסים חוזרים אשר מופיעים ב-time series עבור lag-ים קבועים.

דפוסי עונתיות הם דפוסים נפוצים מאד ונitin למצוא אותם בשל בעיות חיזוי של מכירות, מנויות, מג אויר ועוד. כאשר קיימת עונתיות בתוננים שלנו, המודלים שתיארנו עד כה עלולים להתקשות בניבוי ולסבול מהטיה גדולה שנובעת מהאופי של הנתונים שלא מתישב עם מודל הרוגסיה הילינארית. נסתכל לדוגמה על גרפם המאומתים היומי עליו הדגמנו את החיזוי בעזרת המודלים השונים בפרק זה:



ניתן לראות כי קיימים מקרים מסוימים בגרף המדגים דפוסי עונתיות שבועית (מסומנים באדום). בפרקים הקודמים העלו השערה לפיה העונתיות השבועית נובעת מכך שמש' הבדיקות שבוצעות בסופי שבוע קטן משמעותית בהשוואה לימי חול, וכך גם מספרי המאומתים יורדים בהתאם. כיצד עונתיות זו פוגעת בחיזוי? נניח כי אימנו את המסוג שלנו בעזרת אחד מהמודלים אשר הוצגו עד כה בחלוקת הקודמים. עבור כל אחד מהמודלים, ערך החיזוי ליום כלשהו נקבע לפי קומבינציה ליניארית של פרמטרים המוחשבים על ערכי הדגימות בימים הקודמים. המודל שלנו יוצא מtower נקודת הנחה, כי ערך התוכנה המבוקשת עבור יום t יושפע מהערכים בימים $2 - t$, $1 - t$ וכו'. כתת נניח ואנחנו מנסים לנבא את מס' המאומתים ביום שישי כלשהו אשר נופל באזוריים בהם מודגמת עונתיות ב-time series שלנו. אם ננסה לנבא את מס' המאומתים ביום שישי בהתבסס על מס' המאומתים בימים רביעי וחמישי (לדוגמה), רוב הסיכויים שהחיזוי שלנו יהיה גבוה משמעותית מערך האמת. לעומת זאת, ניתן כי דזוקא אם נסתכל על מס' המאומתים ביום השישי שלפני, נקבל חיזוי מדויק יותר, מכיוון שהוא מtabסס על ה-lag העוני במקומו על ה-lag הרציף.

לאחר שהציגנו את הבעיה איתה אנחנו מתמודדים, נרצה לבצע את השלבים הבאים:

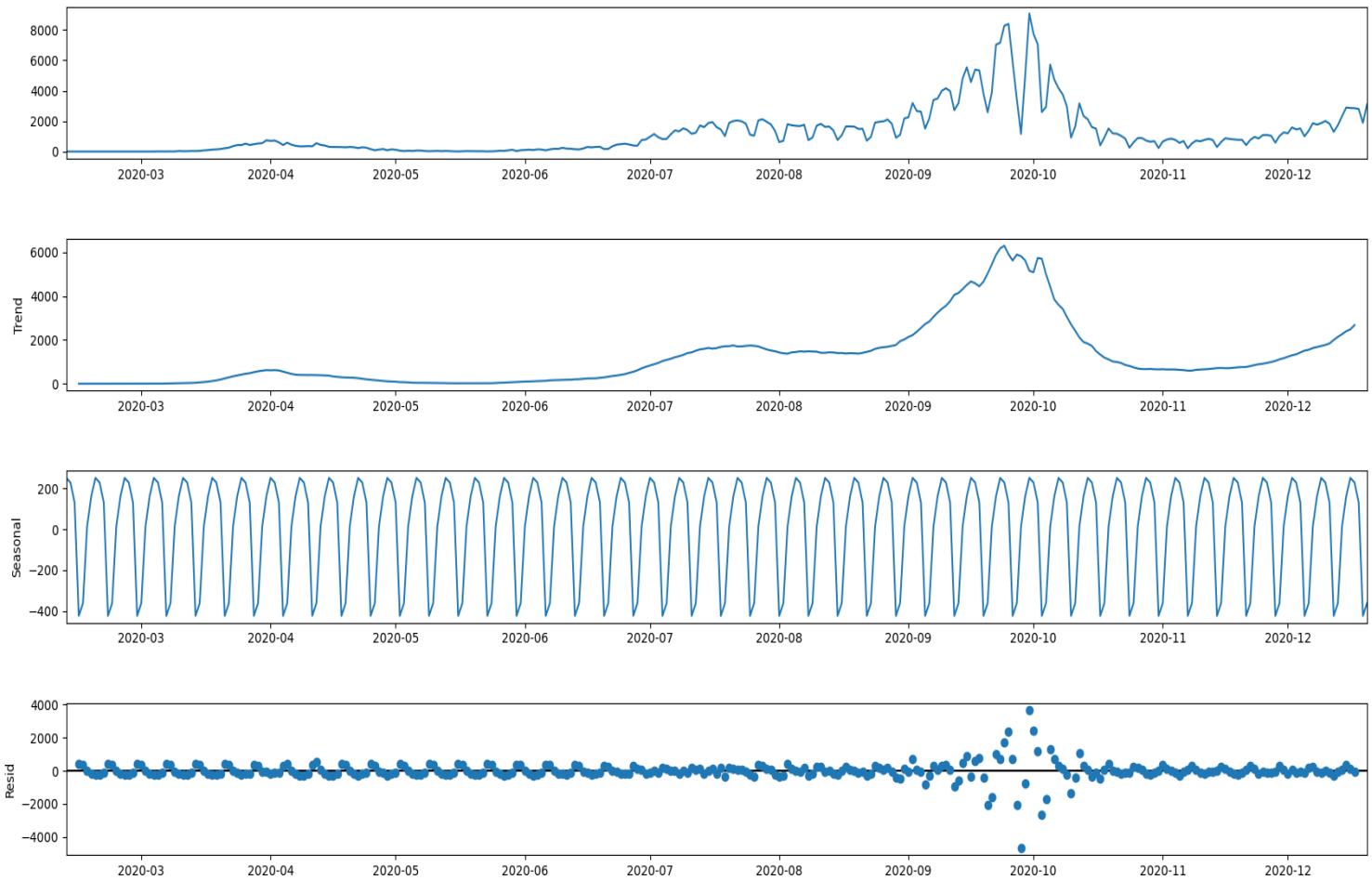
1. זיהוי מדויק של דפוסי עונתיות ב-time series שלנו (seasonal decomposition).
2. הצעת מודל אשר יודע לשקלל את העונתיות בתנאים ולתת חיזוי בהתאם.

Seasonal decomposition – על מנת לקבל תמונה ברורה יותר באשר למגוון עונתיות אשר קיימות לנו ב-time series, נרצה לנסות ולפרק את הפונקציה הראשית שלנו, שהיא ערכיו התכוונה כתלות בזמן, לשילוש פונקציות נפרדות:

1. פונקציית trend – פונקציית ההתקדמות הכללית של התכוונה כתלות בזמן. נסמן פונקציה זו ב- $T(t)$.
2. פונקציית seasonality – פונקציה ציקלית אשר מבטא את העונתיות שקיימת לנו בתנאים. גודל המחזור של הפונקציה יהיה שווה ל-lag העוני של הנtones. נסמן פונקציה זו ב- $S(t)$.
3. פונקציית residuals – פונקציה זו מבטא את הרעשים שאינם בעלי ביטוי בתrend או ב-seasonality. נסמן פונקציה זו ב- $E(t)$.

קיים שני סוגי פירוקים – פירוק חיבורו ופירוק כפלי. עבור פירוק כפלי, ערך התכוונה יתקבל ע"י הכפלת של שלושת הפונקציות: $Y(t) = T(t) \cdot S(t) \cdot E(t)$. עבור פירוק חיבורו, ערך התכוונה יתקבל ע"י חיבור שלושת הפונקציות: $Y(t) = T(t) + S(t) + E(t)$.

על מנת לבצע את הפירוק של התכוונה שלנו לפונקציות היוצרות, נשתמש בפונקציה seasonal_decompose מתוך הספרייה statsmodels. פונקציה זו מפעילה קונבולוציות על הנתונים על מנת לסנן את פונקציה trend, ולאחר מכן מפעילה חישובי ממוצע על כל תקופה זמן על מנת לקבל את המרכיב העוני. נבצע את הפירוק על ה-time series המקורי שלנו ונציג את הגרפים עבור כל אחד מהפונקציות המתקבלות מהפירוק:



מיפוי שקיבלו ניתן לראות בבירור את התוומה של כל אחד מחלקים השונים לפונקציית התכונה שלנו. בפרט, ניתן לראות בבירור את המרכיב העוני בגרף Seasonal . הגרף העוני מציג דפוס מחזורי עם מחזוריות של 7 ימים. בנוסף, ניתן לראות שההבדלים המוצעים כתוצאה מהעוניות הם מאד גדולים ונעים בין -400 ל- $+200$. המשמעות היא שנוכל לקבל הבדל ממוצע של 600 חולים כתוצאה מדפסי העוניות שיש לנו נתונים. מכאן מופיע נספ' שני ניתן לראות מהפירוק הוא העלייה בפונקציית residuals בחודשים ספטember ואוקטובר 2021. עליה זו מסבירה את התנודות החודשיות שיש לנו בתקופה זו בגרף התכונה המקורית שלנו, ויכולות להצביע על כך שקיימות אונומליה כלשהי בתקופה זו. זיהוי אונומליות הוא נושא חשוב שיכל לעזור לנו לצפות מהם האזורים בהם נתקשה בהזיהוי לעומת אזורים אחרים. ניר כי הפירוק בו השתמשנו היה פירוק חיבורו.

על מנת להתמודד עם העוניות time series שלנו נשתמש במודל SARIMA (לרוב נקרא גם בשם ARIMA Seasonal). מודל זה נבנה על גבי מודל ARIMA שהציגו בחלק הקודם, אך מוסיף לו ממד נוסף של עוניות.

מודל SARIMA(p, d, q)(P, D, Q) _{m} מכיל 7 פרמטרים:

- הפרמטרים (p, d, q) הם הפרמטרים עבור חלק של המודל הזהה למודל ARIMA. פרמטרים אלו הם הפרמטרים הלא עוניים (non-seasonal).

- הפרמטרים (P, D, Q) הם הפרמטרים העונתיים (seasonal orders). הפרמטר P הוא הפרמטר עבור המרכיב העונתי של ה-AR, Q הוא הפרמטר עבור המרכיב העונתי של ה-MA, והפרמטר D הוא הפרמטר עבור ה- $diff$ העונתי.
- הפרמטר m הוא הפרמטר המבטא את האינטראול של העונתיות במודל (לדוגמא $-7 = m$ עבור ה- $time series$).

משוואת המודל היא משווהה כפליית (multiplicative seasonal ARIMA) המוצגת באופן הבא :

$$\Phi_P(B^m)\phi_p(B)\nabla_s^D\nabla^d y_t = \Theta_Q(B^m)\theta_q(B)\varepsilon_t$$

כאשר :

- $\Phi_P(B^m)$ הוא האופרטור העונתי מסדר P של מודל ה-AR.
- $\phi_p(B) = (1 - \phi_1B - \dots - \phi_pB^p)$ הוא האופרטור הרגיל מסדר p של מודל ה-AR.
- ∇_s^D מייצג את אופרטור ה- $diff$ העונתי. המשמעות היא שההפרש נלקחים בין כל שתי תקופות זמן הנבדלות ביןיהן במחזור עונתיות אחד או יותר (כטלות בפרמטר D), בняוגד לאופרטור $diff$ הרגיל אשר מחסיר בין תקופות זמן עוקבות.
- ∇^d מייצג את אופרטור ה- $diff$ הרגיל.
- $\Theta_Q(B^m) = (1 - \Theta_1B^m - \dots - \Theta_QB^{mQ})$ הוא האופרטור העונתי מסדר Q של מודל ה-MA.
- $\theta_q(B) = (1 - \theta_1B - \dots - \theta_qB^q)$ הוא האופרטור הרגיל מסדר q של מודל ה-MA.
- ε_t מייצג את תהליך השגיאה (white noise process).

הערה : האופרטור B הוא אופרטור backshift. זהו האופרטור שימושי לעובדה עם time series lags. כדי לסמן את ההפרש מסדר d נוכל נרשום $y_t = (1 - B)^d y_{t-1}$.

בחירה הפרמטרים עבור מודל SARIMA

בחירה הפרמטרים עבור מודל SARIMA מתחלקת לכמה חלקים :

- פרמטרים עבור ARIMA – הפרמטרים q, d, p ייבחרו כפי שהדגמנו בחלוקת הקודמים, בעזרת בחינה של מידת הסטציונריות של ה-time series שלנו, ובעזרת פונקציות ACF ו-PACF. נשים לב שכעת נוכל לבחור ורק את ה-lags הקרובים ביותר התוצאה אינה זניחה מבחינה סטטיסטית. בחלוקת הקודמים בחרנו ב-lags גובהים יותר (לדוגמא $7 = q = d = 7$) על מנת "להרוויח" גם את המרכיבים העונתיים, אך שילמנו על כך בעובדה שהמודל שלנו כולל גם lags זניחים שפוגעים בחיזויו.
- הפרמטר m ייבחר לפי הנитוח שעשינו נתונים בעזרת decomposition-seasonal.

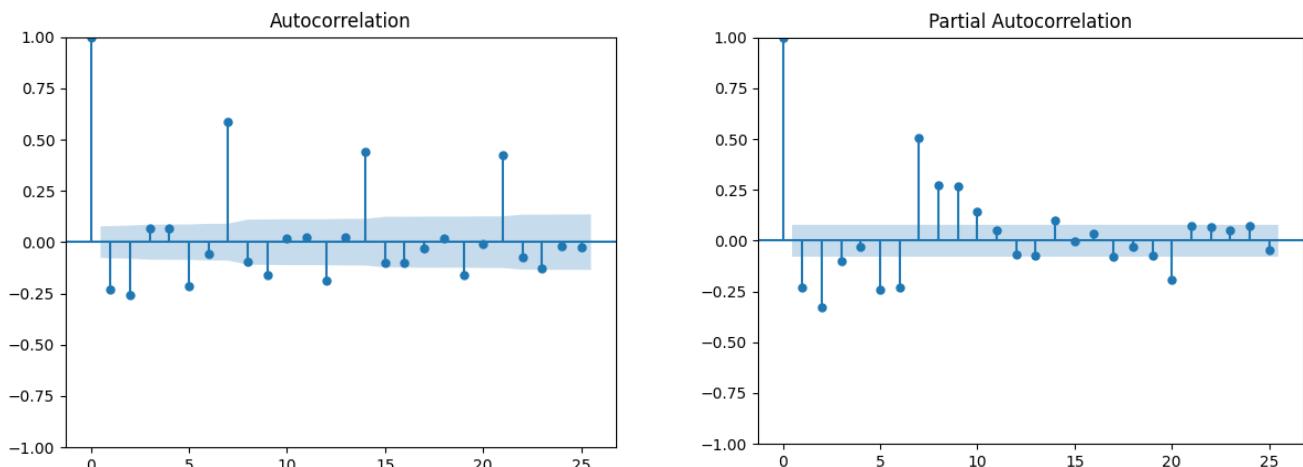
- הפרמטרים P , Q עבור המרכיב העונתי ייבחרו בהתאם לדפוסי הקורלציה בין המרכיבים העונתיים D -ב-ACF ו-PACF. הפרמטר D עוזר בהשגת סטציונריות עונתית. לרוב מספיק יהיה לבחור $D = 1$ על מנת להשיג את האפקט המבוקש.

חיזוי באמצעות SARIMA

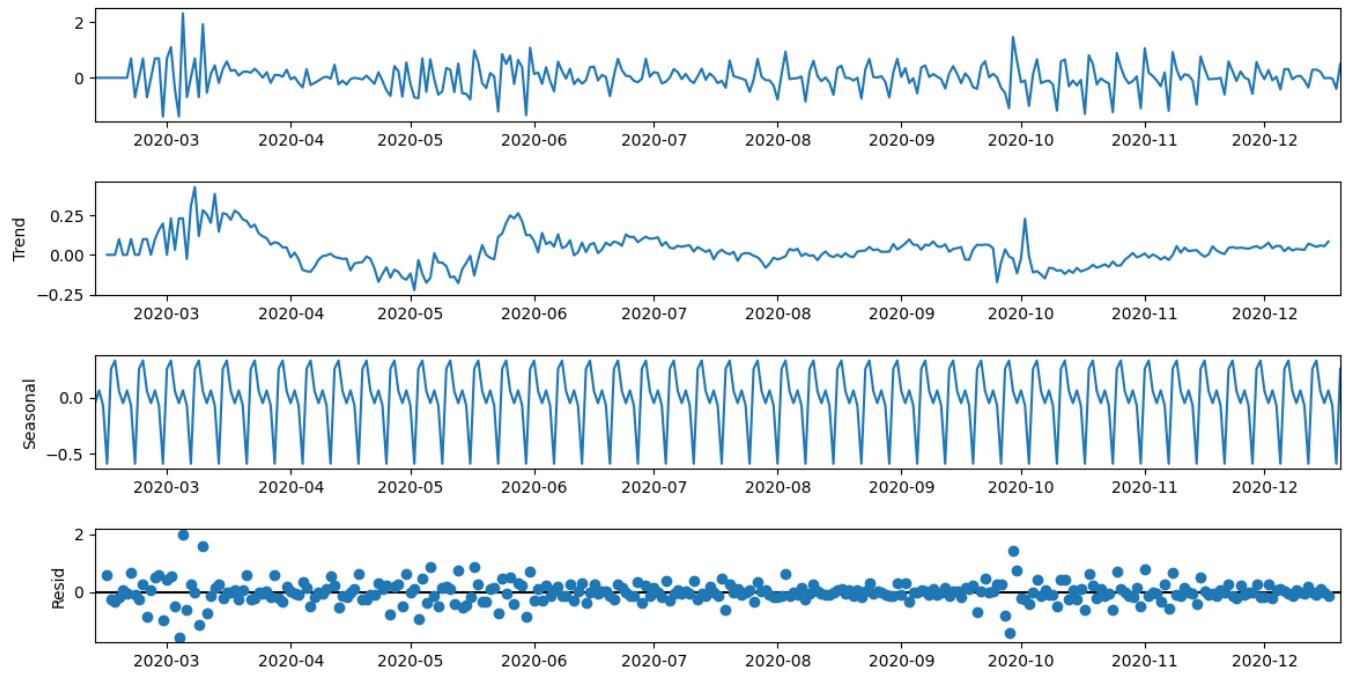
לאחר בחירת הפרמטרים עבור המודל, יבוצע תהליך מיידה בו יבחרו הפרמטרים עבור משווהת החיזוי. לאחר מכן נוכל לבחור את התקופה אותה נרצה לחזות.

נראה כעת דוגמה בסיסית לחיזוי באמצעות מודל SARIMA. לצורך הדוגמה השתמש שוב ב-time series שהוצג בתחילת הפרק, המציג את מס' המאומטים ברמה הלאומית. השתמש באותו הטרנספורמציות שביצעו על ה-*data*-*df* כדי להביא אותו למצב סטציוני. תהליך האימון יבוצע עד התאריך ה-20/12/2020 (בדומה לרוב הדוגמאות מהחלקים הקודמים).

זכור את פונקציות ה-ACF וה-PACF שנקלל לאחר הפעלת הטרנספורמציות על ה-*data* :



עבור המרכיב הלא-עונתי – ניתן לראות כי שני הגרפים דועכים אל האזורי הזוני שלlag השני, ולכן נבחר $p = 2$, $q = 2$. על מנת לקבוע את הפרמטר העונתי, נפעיל את פונקציית ה-*seasonal decomposition* על מנת לאשש שכן ישנו דפוס עונתי ולקבוע את האינטראול העונתי :



כפי שניתן לראות, גם אחרי הפעלת הטרנספורמציה עדין קיבלנו מרכיב עונתי ברור בנתונים שלנו, ולכן נקבע את הפרמטר העונתי שלנו להיות $7 = m$. עבור הפרמטרים העונתיים נבחר מודל פשוט: $P = 1$, $D = 1$, $Q = 1$. המשמעות היא שבמעבר כל חייזי ניעזר ברגression לינארית המבוססת על מחזור עונתי אחד אחרה (עבור MA ו-AR) ובנוסף נבצע $diff$ עונתי של מחזור אחד.

נראה כיצד נראה מודל $SARIMA(2,0,2)(1,1,1)_7$ ומבצע חייזי של מס' ימים עתידיים. פلت האימון של המודל:

```

Model Fitting Time: 1.152909517288208
SARIMAX Results
=====
Dep. Variable: daily_new_cases   No. Observations: 313
Model: ARIMA(2, 0, 2)x(1, 1, [1], 7) Log Likelihood: -101.377
Date: Fri, 31 Dec 2021   AIC: 216.754
Time: 18:42:30   BIC: 242.819
Sample: 02-12-2020   HQIC: 227.178
                    - 12-20-2020
Covariance Type: opg
=====

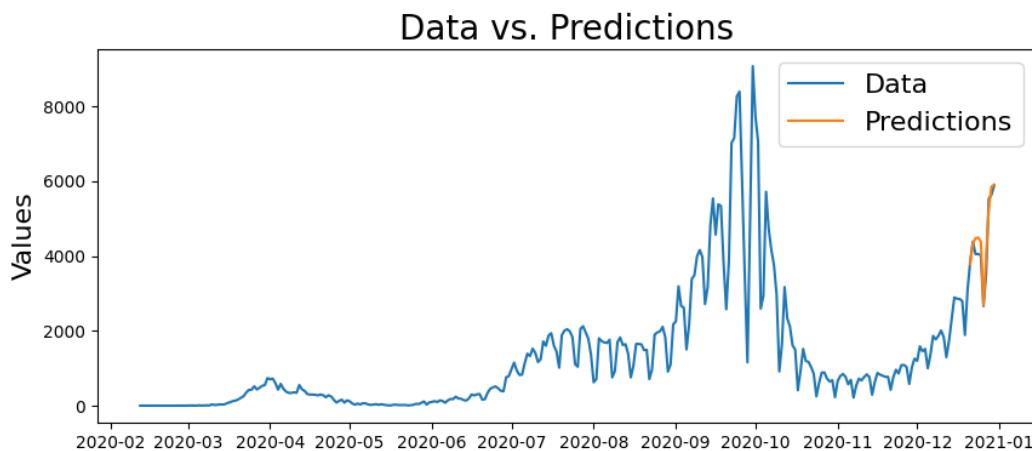
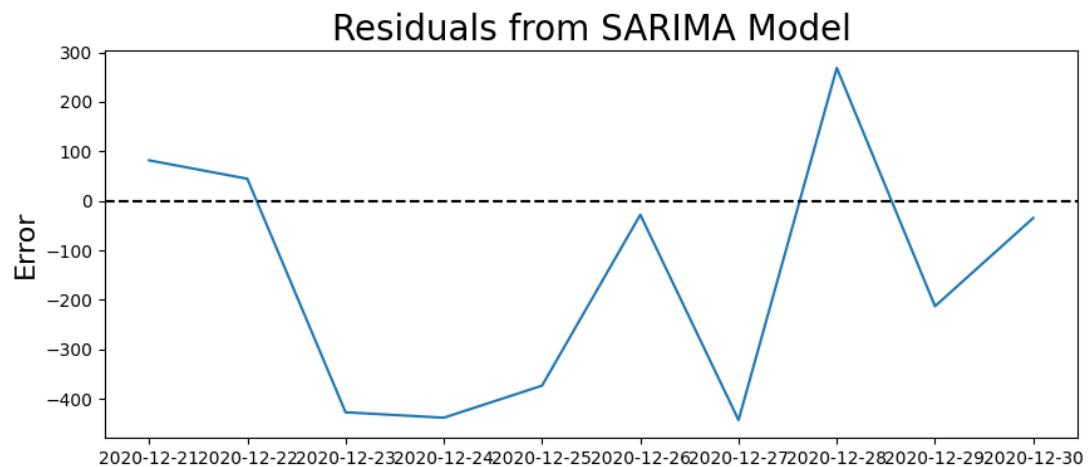
            coef    std err        z     P>|z|      [0.025      0.975]
-----
ar.L1    -0.7382    0.099    -7.475    0.000    -0.932    -0.545
ar.L2    -0.2879    0.083    -3.462    0.001    -0.451    -0.125
ma.L1     0.2397    0.114     2.102    0.036     0.016     0.463
ma.L2    -0.1702    0.103    -1.652    0.098    -0.372     0.032
ar.S.L7    0.4549    0.048     9.560    0.000     0.362     0.548
ma.S.L7   -0.9740    0.070   -13.915    0.000    -1.111    -0.837
sigma2     0.1085    0.008    14.033    0.000     0.093     0.124
=====

Ljung-Box (L1) (Q):      0.01  Jarque-Bera (JB):      203.52
Prob(Q):                0.93  Prob(JB):                  0.00
Heteroskedasticity (H):  0.42  Skew:                      0.08
Prob(H) (two-sided):    0.00  Kurtosis:                 6.99
=====
```

- המקדים עם התחלית 'L.ar' הם המקדים של מודל ה-AR הרגיל.
- המקדים עם התחלית 'L.ma' הם המקדים של מודל ה-MA הרגיל.
- המקדים עם התחלית 'S.L.ar' הם המקדים של מודל ה-AR העונתי.
- המקדים עם התחלית 'S.L.ma' הם המקדים של מודל ה-MA העונתי.

בנוסף ניתן לראות מבחן של ערכי ה- AIC כי כמעט וכל המקדים של המודל אינם זמינים מבחינה סטטיסטית, דבר המרמז על כך שהפרמטרים שבחרנו עבור המודל היו מתאימים עבור הנתונים שלנו.

עתה ניתן לבצע חיזוי באמצעות המודל המאומן. לצורך הדוגמה נחוצה כעט תקופה של 10 ימים, החל מסוף תקופת האימון. נציג כפלו את גרפ' השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:



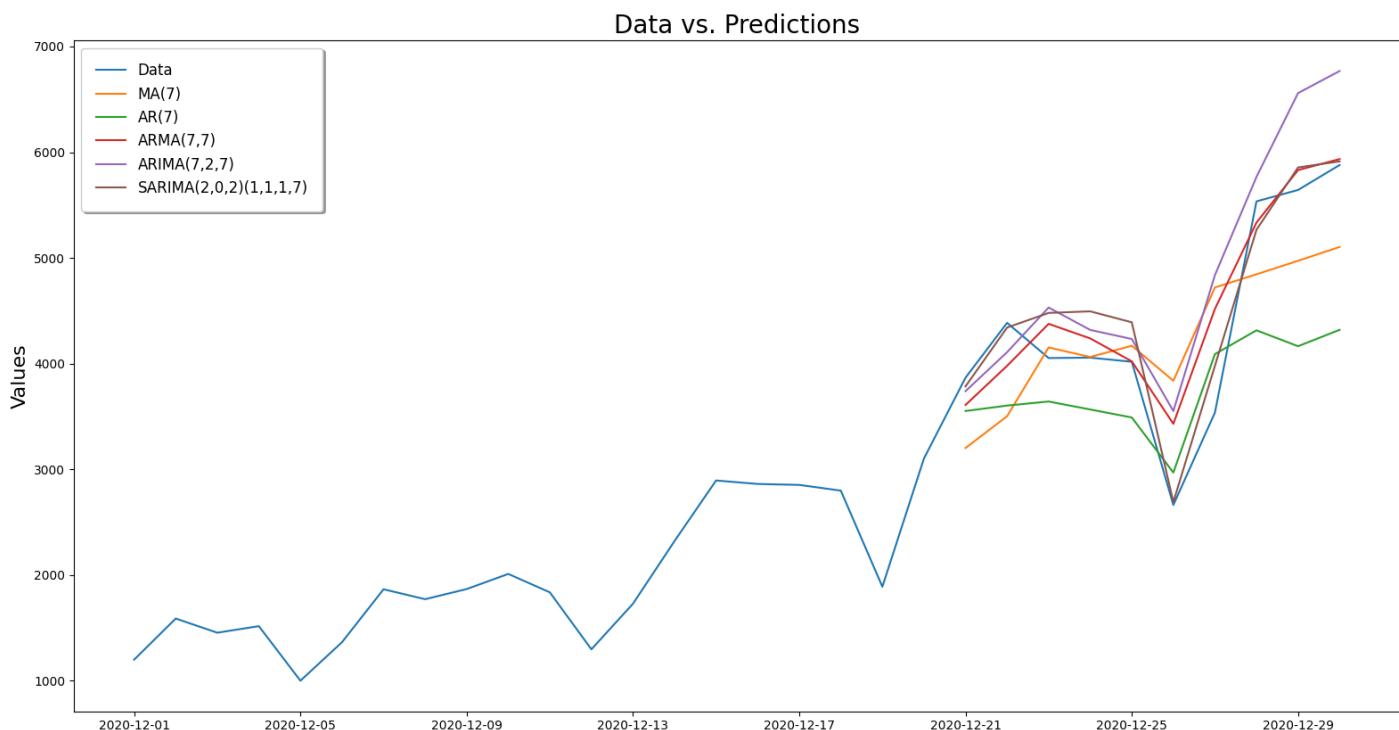
```
##### Metrics for model accuracy #####
Test Data Mean: 4363.799999999999
Mean Absolute Error 235.1105339815706
Mean Squared Error: 83740.51269219557
Mean Absolute Percentage Error: 0.05652988199048416
Root Mean Squared Error: 289.37953053420273
Median Absolute Error: 240.59555309060852
R2 score: 0.9112712981039852
```

נציג את המטריקות השונות עבור דיקוק המודל:

ניתן לראות כי קיבלנו תוצאות חיזוי מרשימות – אחוז השגיאה עומד על כ-0.05, תוחלת השגיאה עומדת על כ-235.1 בהשוואה לתוחלת ערכי ה-test set העומדת על כ-4363.8. לפי מטריקת R^2 קיבלנו כעט דיקוק של 0.91.

השוואת מודלים

לאחר שהציגנו את כל המודלים השונים, נציג השוואת תוצאות החיזוי של כל אחד מהמודלים השונים על אותה תקופה זמן. כל המודלים ישתמשו ב-time series שהוצע בתחילת הפרק. הפרמטרים עבור כל מודל ייקבעו בהתאם לאנליזות שביצעו במהלך כל חלק. תהליך האימון יבוצע עד התאריך ה-20/12/2020, וממנו נחזה תקופה של 10 ימים עד ה-30/12/2020. נציג את תוצאות החיזוי בגרף (צמצמנו את טווח התאריכים המוצג בגרף כדי להציג את ההבדלים בין המודלים השונים) :



נציג את המטריקות הנבחנות עבור כל מודל בטבלה :

	MA	AR	ARMA	ARIMA	SARIMA
<i>R2 score</i>	0.41	0.16	0.78	0.51	0.91
<i>MAPE</i>	0.15	0.16	0.09	0.13	0.05

סיכום

מודל SARIMA חותם את פרק זה ומהווה שילוב של כל העקרונות שראינו לאורך הפרק. ראיינו כיצד שילוב של מודלים שונים של גרסיה לינארית על סמך פרדיקטים שונים, בשילוב עם התשבות במרקיבים העונתיים, מצליחים להביא לבסוף למודל בעל אחזוי דיקט מרשימים מאד. לאורך כל הפרק שמננו דגש רב על הבנה וכונה של המודלים איתם אנחנו עובדים, לרבות ניתוח סטציונריות, עונתיות ודפוסי קורלציה שונים. הבנה של העקרונות הללו מאפשרת לנו לבחור פרמטרים מתאימים, לאמן ולהזות באמצעות המודלים שלנו ולהגיע לאחזוי דיקט טובים, כל זאת ללא ביצוע ניסויים מפורשים לאמון המודל ו/או כיוונו פרמטרים אוטומטי.

בפרק הבא נבצע ניסויים והדגמות, ונציג אופטימיזציות לשיפור החיזוי שלנו באמצעות המודלים שהוצעו בפרק זה.

הדגמות וניסויים

בפרק הקודם הצגנו את האלגוריתמים השונים המאפשרים לנו לבצע חיזוי של ערכים עתידיים. הרأינו את המודל התיאורטי אשר עומד מאחורי כל אחד מאלגוריתמי הלמידה, וכמו כן הצגנו שלל דרכים אשר עוזרים לנו לקבוע את הפרמטרים המתאימים עבור כל אחד מהאלגוריתמים. בפרק זה נתמקד פחות בצד התיאורטי, וננסה להביא את האלגוריתמים שלנו לביצועים הכי טובים בעזרת כיווןן פרמטרים, חידושים ואופטימיזציות אלגוריתמיות ואולי טרנספורמציה לעיבית חיזוי חדשה.

הנושאים בהם עוסוק בפרק זה :

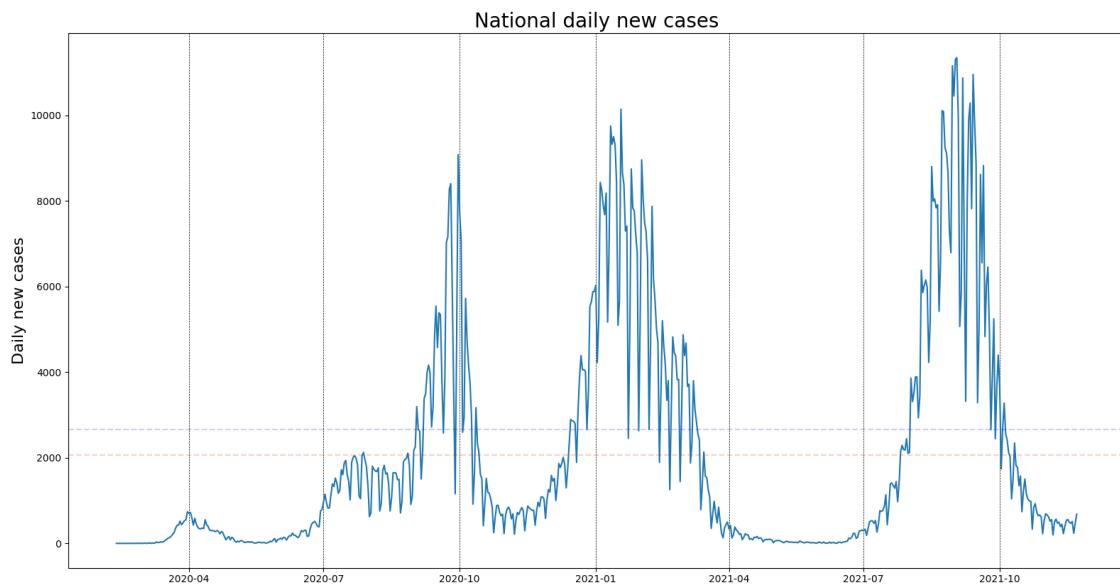
1. Auto ARIMA
2. Rolling forecast
3. Rolling average smoothing

הערה : לאורך פרק זה נסתמך על תוצאות ומסקנות אליהן הגיעו בפרקים הקודמים בנושאים הקשורים לניתוח הנתונים וקביעת פרמטרים עבור המודלים השונים שלנו.

Auto ARIMA

בסוף הפרק הקודם הצגנו את מודל SARIMA, שהוא מודל שמכיל בתוכו את כל האלמנטים השונים שראינו לאורך הפרק – מרכיב AR – (auto regression), מרכיב של MA (moving average), מרכיב לטיפול בסטציונריות של הנתונים (integrated) ומרכיב לטיפול בעונתיות (stationary). מודל SARIMA המלא מוצג באופן הבא : $SARIMA(p, d, q)(P, D, Q)_m$ ומכיל 7 פרמטרים שונים של המודל. הרأינו כיצד בעזרת ניתוח של קורלציות בין-lags השונים (בעזרת פונקציות ACF וה-PACF) ופירוק המרכיבים העונתיים של הפונקציה הראשית שלנו (seasonal decomposition), ניתן להסיק את הפרמטרים המתאימים עבור המודל. למרות כל האמור לעיל, יש לזכור כי כל הכללים שהציגו (ועוד כלים רבים אחרים שלא הציגו) מהווים מעין יוריסטיקה אשר אמורה לעזור לנו בבחירה הפרמטרים האופטימליים, אך אם נרצה לקבל את התוצאות האופטימליות עבור הנתונים שלנו, אין לנו ברירה אלא לבצע כיוונו פרמטרים. עבור מודל ה-SARIMA, קביעת משווהת החיזוי האופטימלית תדריש כיוונו פרמטרים עבור 7 פרמטרים שונים. מシימה זו היא משימה לא פשוטה כל וודשת משאבים חישוביים כבדים מאד. לשם כך נוכל להיעזר בכלי Auto ARIMA. כדי זה עוזר לנו לבצע כיוונו פרמטרים עבור מודלים ממשפחת SARIMA.

כלי הכוונו שלנו מגיע בתצורת פונקציה בשם auto_arima (מתוך הספרייה statsmodels). פונקציה זו מציעה שלל של אפשרויות אותן נסקר בחלק זה. ראשית, נתען את ה-time series אליו נעובד. נעבד עם ה-time series המציג את רמת המאומתים ברמה הלאומית. בנגד לפרק הקודם, הפעם נתען את הנתונים עד לתאריך 22/11/2021. נציג את ה-time series בצורה גרפית :



כפי שניתן לראות, גורף זה מדגים את ארבעת גלי התחלואה העיקריים שהיו במדינת ישראל מפוץ המגיפה ועד לחודש נובמבר 2021.

נרצה להתחיל לעבוד עם הכללי `auto arima` על מנת להבין מהו המודל האופטימלי עבור הנתונים שלנו. תחילה, נרצה להבין כיצד נוכל להגיד שמודל א' יותר טוב ממודל ב'? כדי לענות על שאלה זו, כלים כגון `auto arima` (וכלים רבים נוספים) משתמשים בקריטריונים מסוימים לפיהם ניתן לאמוד את השגיאות של המודל. במסגרת פרויקט זה לא נסקור את כל הקריטריונים השונים, אך נציג את הקריטריון הנפוץ ביותר בו השתמש להערכת המודלים שלנו.

Akaike information criterion (AIC) – קритריון זה משמש להערכת השגיאות של מודלים סטטיסטיים, ובכך לתת אומדן לאיכות המודל. הקритריון בנוי על עקרונות מתורת האינפורמציה, ומעיריך את אובדן המידע היחסי של המודל. ככל שאובדן המידע קטן יותר, כך המודל נחשב לטוב יותר. השאיפה שלנו בהתקמת המודל תהיה להקטין את ערך ה-AIC ככל הנitin.

כעת נוכל להתחיל לבחש את המודל הטוב ביותר עבור הנתונים שלנו באמצעות שימוש ב-`auto arima`. כפי שהציגנו לעיל, באמצעות כליה זה ניתן להריץ מס' רב של ניסויים על הנתונים שלנו, כאשר המטרה הסופית שלנו היא לקבל את הפרמטרים הטוביים ביותר עבור מודל-SARIMA. כפי שראינו, חיפוש ממצאה על מרחב כל האפשרויות עלול להיות משימה מאד קשה חישובית, לכן הפונקציה משתמשת בפרמטרים רבים שהמשתמש קבוע על מנת לעוזר ולצמצם את מרחב החיפוש. ככל שנקבע את הפרמטרים הללו בזורה טובה יותר תוך היכרות עם הנתונים שלנו, כך נוכל להגדיל את הסיכויים למצוא את המודל הטוב ביותר.

על מנת להציג את השימוש בפונקציה `auto sarima`, נתחל מהרצת הפונקציה על ה-time series שלנו בקורס הנאיבית ביותר. ההרצאה הנאיבית נעשית על ידי הזנה של הנתונים לתוך הפונקציה, ללא קביעה של אף פרמטר נוסף מהפרמטרים שהפונקציה מציעה לנו. במצב זה, הפונקציה תשתמש בערכי ברירת המחדל שלא כדי לתת לנו תוצאה בזמן סביר.

נ裏ץ את הפונקציה `auto arima` ונדריס את הפלט של הפונקציה. הפלט של הפונקציה מציג את המודל הנבחר, כלומר זה שהצליח לモען את קритריון הבחירה (במקרה שלנו את קритריון ה-AIC):

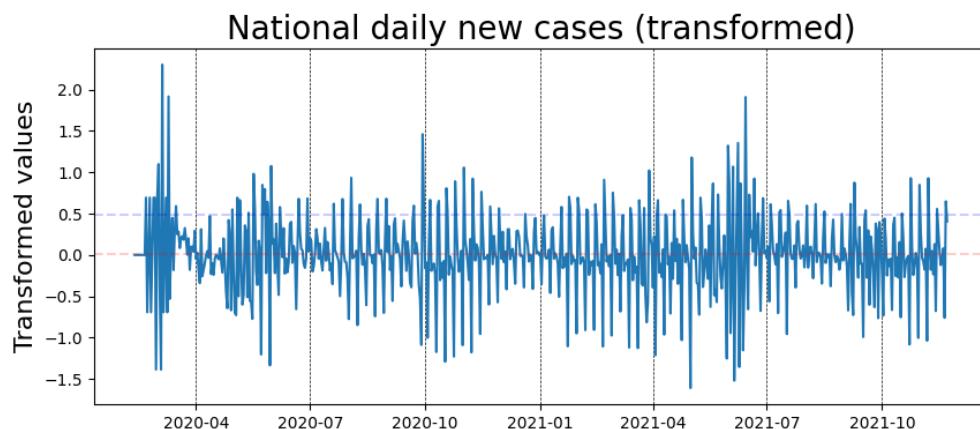
Auto ARIMA runtime: 10.051553726196289 seconds SARIMAX Results						
Dep. Variable:	y	No. Observations:	650 <th data-cs="3" data-kind="parent"></th> <th data-kind="ghost"></th> <th data-kind="ghost"></th>			
Model:	SARIMAX(5, 1, 2)	Log Likelihood	-5265.193 <th data-cs="3" data-kind="parent"></th> <th data-kind="ghost"></th> <th data-kind="ghost"></th>			
Date:	Sat, 01 Jan 2022	AIC	10546.387			
Time:	12:34:22	BIC	10582.190			
Sample:	0 - 650	HQIC	10560.275			
Covariance Type:	opg					
	coef	std err	z	P> z	[0.025	0.975]
ar.L1	0.3750	0.047	7.978	0.000	0.283	0.467
ar.L2	-0.8460	0.029	-28.812	0.000	-0.904	-0.788
ar.L3	-0.1524	0.037	-4.114	0.000	-0.225	-0.080
ar.L4	-0.2793	0.024	-11.474	0.000	-0.327	-0.232
ar.L5	-0.3919	0.026	-15.036	0.000	-0.443	-0.341
ma.L1	-0.7323	0.042	-17.231	0.000	-0.816	-0.649
ma.L2	0.6068	0.034	17.781	0.000	0.540	0.674
sigma2	6.793e+05	1.75e+04	38.907	0.000	6.45e+05	7.14e+05
Ljung-Box (L1) (Q):		0.01	Jarque-Bera (JB):	3157.92		
Prob(Q):		0.91	Prob(JB):	0.00		
Heteroskedasticity (H):		17.82	Skew:	-1.28		
Prob(H) (two-sided):		0.00	Kurtosis:	13.50		

נמתח את התוצאות שקיבנו :

- המודל הנבחר הוא $ARIMA(5,1,2)$ (מסומן בכתום). נזכיר כי הזנו את time series שלנו אל הפונקציה ללא הפעלת טרנספורמציות כלל, לכן לא מפתח שנבחר פרמטר $d = 1$ על מנת להביא את time series למצב סטטיזוני.
- ציון ה-AIC עבור המודל שלנו הוא 10546.387 (מסומן באדום).
- ניתן לראות את המקדים שנבחרו עבור המודל ואת ערך ה- k עבור כל אחד מהמקדים (מסומן בירוק). כל המקדים שנבחרו לא זניחים מבחינה סטטיסטית ומשמעותם על משווהת החיזוי.
- זמן הריצה של הפונקציה עמד על כ-10 שניות. זמן הריצה עשוי להשנות כתלות בחומרה המרכיב, אך עדיין נוכל להשתמש בהפרשי הזמן כדי להציג את ההבדל בין ה的信任ות (כל הניסויים הורצו על אותו המחשב).

נשים לב שהנתונים שהזנו לתוך הפונקציה הם הנתונים הגולמיים שלנו שלא עברו אף טרנספורמציה. פונקציית `auto arima` יודעת להתמודד עם `data` לא סטטיזוני, אך היא עשויה זאת באמצעות שימוש בטרנספורמציה ה- $diff$ בלבד (החלק ה-integrated במודול SARIMA). מכיוון שיש לנו ספרייה שלמה המותאמת לביצוע טרנספורמציות, השלב הראשון שנוכל לעשות על מנת לשפר את הביצועים של `auto`

הוא לספק לפונקציה נתונים שכבר הבנו במצב סטציוני. לשם כך נפעיל את טרנספורמציות ה-`arima` וה-`diff` כפי שעשינו בחלק הקודם ונקבל את הטרנספורמציה הבאה:



כפי שניתן לראות מהרצת ה-`ADF test`, כתה הנתונים שלנו מציגים רמת סטציונריות גבוהה.

```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -4.390343
P-Value                  0.000309
# Lags Used              20.000000
# Observations Used     629.000000
Critical Value (1%)      -3.440789
Critical Value (5%)      -2.866146
Critical Value (10%)     -2.569223
dtype: float64
Is the time series stationary? True
```

נريץ שוב את פונקציית `auto_arima` הנאייבית על ה-time series שלנו שuber טרנספורמציות. הפעם, מלבד הנתונים עצם, נספק לפונקציה `auto_arima` את הדגל `stationary=True`. דגל זה מסמל שה-`data` שמדובר בה הוא `stationary`, ונמצא במצב סטציוני, וכן הפונקציה לא מבזבזת משאבים על חישוב הסטציונריות וביצוע טרנספורמציות על מנת להביא את ה-`data` במצב סטציוניiri בעצמה (כפי שעשתה בהרצאה הקודמת).

נריץ את הפונקציה `auto_arima` ונדפיס את הפלט של הפונקציה:

```
Auto ARIMA runtime: 8.743702173233032 seconds
SARIMAX Results
=====
Dep. Variable:                      y    No. Observations:                 650
Model:             SARIMAX(1, 0, 3)   Log Likelihood:            -339.653
Date:                Sat, 01 Jan 2022   AIC:                   689.306
Time:                    14:17:36       BIC:                   711.691
Sample:                   0 - 650    HQIC:                  697.988
Covariance Type:            opg
=====
              coef    std err        z     P>|z|      [0.025    0.975]
-----
ar.L1      0.8014    0.040    20.035      0.000      0.723     0.880
ma.L1     -1.3033    0.047   -27.871      0.000     -1.395    -1.212
ma.L2      0.0805    0.068     1.190      0.234     -0.052     0.213
ma.L3      0.3873    0.039     9.833      0.000      0.310     0.464
sigma2     0.1661    0.006    25.601      0.000      0.153     0.179
=====
Ljung-Box (L1) (Q):                  0.01    Jarque-Bera (JB):           196.15
Prob(Q):                           0.92    Prob(JB):                     0.00
Heteroskedasticity (H):               1.10    Skew:                         -0.63
Prob(H) (two-sided):                  0.50    Kurtosis:                      5.37
=====
```

ננתן את התוצאות שקיבלו :

- המודל הנבחר הוא $ARIMA(1,0,3)$. ניתן לראות כי בעת נבחר מודל אחר בהשוואה להרצתה הקודמת. בנוסף ניתן לראות כי בעת לא נבחר פרמטר עבור החלק-h-integrated ($d = 0$). זה הגיוני מכיוון שהזנו למודל data שכבר נמצא במצב סטטיסוני.
- ציון AIC של המודל הינו 689.306. חשוב להזכיר כאן כי אמנים קיבלו ציון AIC נמוך משמעותית מההרצתה הקודמת, אך שתי התוצאות הללו הן לא ברות השוואה! זאת מכיוון שציון ה-AIC מתייחס לאובדן המידע היחסי בהתייחס לנ נתונים שהזנו למודל. מכיוון שהזנו בעת נתונים בעבר טרנספורמציה, ערכי הנתונים שלנו נמכרים בכמה סדרי גודל מעריכי הנתונים ב-time series המקורי שלנו, ולכן גם ציון AIC נמוך בהתאם.
- זמן הריצה של הפונקציה בעת עומדת על קצר פחתות מ-9 שניות. ניתן ליחס את הירידה זו לעובדה ש"טרמנו" מידע לפונקציה שיחסן ממנה עבודה נוספת.

כפי שציינו, עד כה הרצנו את הפונקציה `auto arima` בצורה הנאיבית ביותר, תוך שימוש בערכי ברירת המחדל עליהם אין לנו שליטה. בעת נגדיר ערכים מפורטים עבור הפרמטרים השונים בהם הפונקציה מאפשרת לנו לשולט, במטרה להגיע לפلت טוב יותר עבור המודל שלנו. נציג כי נציגים כאן רק חלק מהאפשרויות השונות של הפונקציה (קיימות עוד שלל אפשרויות אחרות לא נסקור כאן). נשתמש באותו הזמן עם הטרנספורמציות שהצינו לעיל, ועתה נקרא לפונקציה `auto arima` עם הפרמטרים הבאים :

- `p_max` – פרמטר זה מגביל את הפרמטר `p` המקסימלי אותו ננסה לבדוק. מניתוח שעשינו לפונקציית PACF בפרק הקודם, נבחר קבוע ערך זה להיות $7 = p$.
- `q_max` – פרמטר זה מגביל את הפרמטר `q` המקסימלי אותו ננסה לבדוק. מניתוח שעשינו לפונקציית ACF בפרק הקודם, נבחר קבוע ערך זה להיות $7 = q$.
- `P_max` – פרמטר זה מגביל את הפרמטר `P` (הפרמטר עבור החלק העונתי של AR) המקסימלי אותו ננסה לבדוק. נבחר קבוע ערך זה להיות $4 = P$.
- `Q_max` – פרמטר זה מגביל את הפרמטר `Q` (הפרמטר עבור החלק העונתי של MA) המקסימלי אותו ננסה לבדוק. נבחר קבוע ערך זה להיות $4 = Q$.
- `D_max` – פרמטר זה מגביל את הפרמטר `D` (הפרמטר עבור ה-seasonal differencing) המקסימלי אותו ננסה לבדוק. נבחר קבוע ערך זה להיות $2 = D$.
- `max_order` – פרמטר זה קובע את הסדר המקסימלי שאליו ניתן להגיע עבור החלקים של ה-AR וה-MA במודל שלנו, כולל את הערך המקסימלי עבור $q + p$. מכיוון שהגדרכנו ידנית את ערכי המקסימים עבור כל אחד מהפרמטרים הללו, נוכל לקבוע את הפרמטר הזה להיות `None`, ואז הפונקציה תנסה את כל הקומבינציות האפשריות עבור מודלים תחת הגדרות ה-`max` שנקבעו לכל פרמטר. קביעה זו אמנים יודעים כי זמן זה יהיה חסום ובסיומו נקבל תוצאה לאחר סריקה של כל מרחב האפשרויות הרלוונטי עבור הבעיה שלנו.

- כפי שהסביר לעיל, נקבע דגל זה להיות True מכיוון שאנו יודעים שהנתונים שהזנוstationary לפונקציה כבר הובאו למצב סטציוני, ובכך אנחנו חוסכים זמן בכוונו פרמטרים נוספים.
- דגל זה מסמן האם קיימת עונתיות בתנאים שלנו. מכיוון שאנו יודעים שקיימת עונתיות (מהניתה seasonal decomposition שביצענו בפרק הקודם) נקבע ערך זה להיות True.
- m – ערך זה מסמל את lag העוני. נקבע ערך זה להיות 7 = m (מחזר עונתיות של 7 ימים, כפי שראינו בפרק הקודם).
- אופציה זו מאפשרת לשנות על הקритריון לפיו נבחר את טיב המודל. כפי שהצגנו בתחילת חלק זה, אנחנו משתמש בקריטריון AIC (זהה גם בירית המבחן של הפונקציה).

נ裏ץ את הפונקציה auto arima לאחר התאמת הפרמטרים ונדפיס את הפלט של הפונקציה :

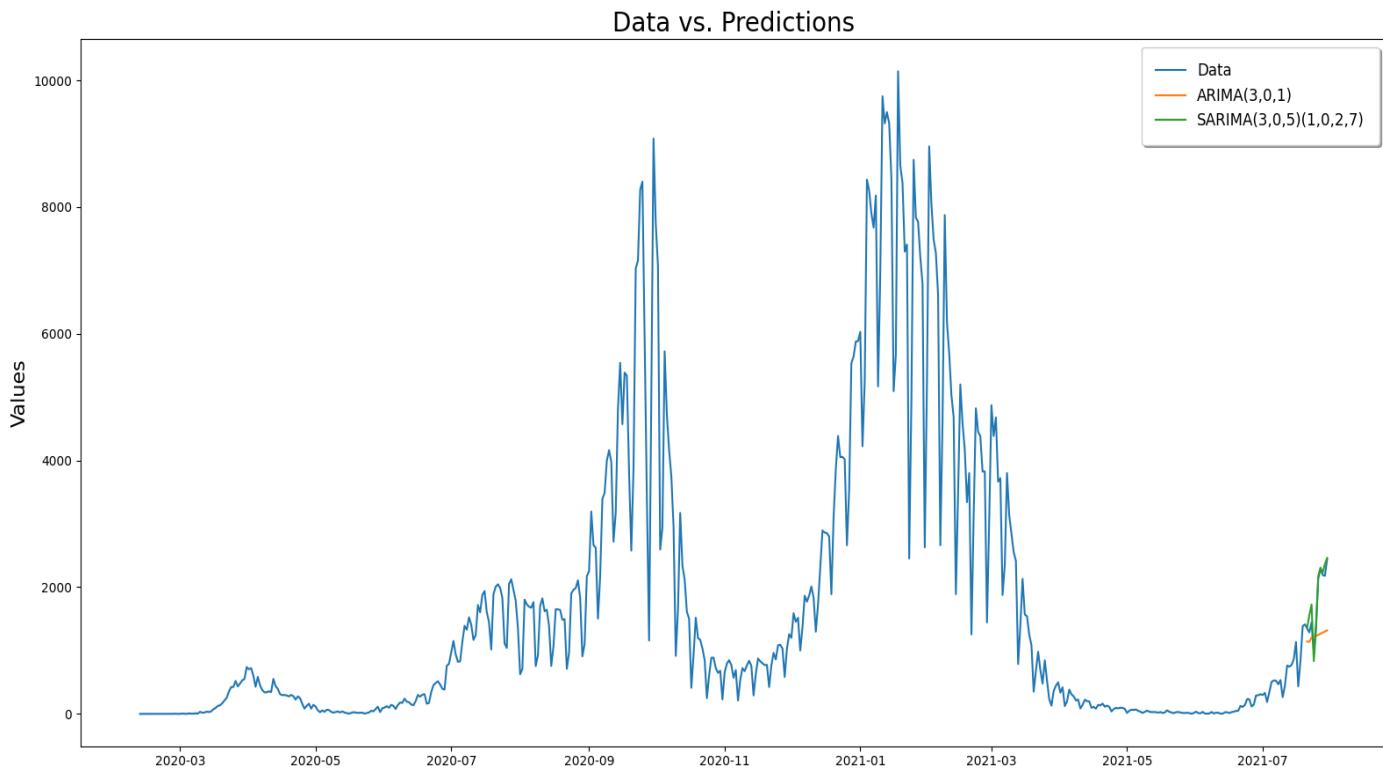
```
Auto ARIMA runtime: 422.72444581985474 seconds
SARIMAX Results
=====
Dep. Variable: y No. Observations: 650
Model: SARIMAX(3, 0, 5)x(1, 0, [1, 2], 7) Log Likelihood: -169.622
Date: Sat, 01 Jan 2022 AIC: 365.245
Time: 14:24:39 BIC: 423.445
Sample: 0 HQIC: 387.819
- 650
Covariance Type: opg
=====
            coef    std err      z   P>|z|      [0.025    0.975]
-----
intercept  7.612e-05  0.000    0.220    0.826    -0.001    0.001
ar.L1       0.0496   0.145    0.342    0.732    -0.234    0.333
ar.L2      -0.0383   0.137   -0.281    0.779    -0.306    0.229
ar.L3       0.7267   0.130    5.573    0.000    0.471    0.982
ma.L1      -0.5684   0.150   -3.777    0.000    -0.863   -0.273
ma.L2      -0.0699   0.172   -0.407    0.684    -0.407    0.267
ma.L3      -0.6566   0.192   -3.425    0.001    -1.032   -0.281
ma.L4       0.4846   0.071    6.838    0.000    0.346    0.624
ma.L5       0.0876   0.057    1.546    0.122    -0.023    0.199
ar.S.L7     0.9802   0.009  110.261    0.000    0.963    0.998
ma.S.L7    -0.6800   0.038  -17.863    0.000    -0.755   -0.605
ma.S.L14   -0.1910   0.040   -4.763    0.000    -0.270   -0.112
sigma2      0.0971   0.003   31.369    0.000    0.091    0.103
=====
Ljung-Box (L1) (Q): 0.11 Jarque-Bera (JB): 697.99
Prob(Q): 0.74 Prob(JB): 0.00
Heteroskedasticity (H): 0.94 Skew: -0.01
Prob(H) (two-sided): 0.67 Kurtosis: 8.08
=====
```

נמתה את התוצאות שקיבלו :

- המודל הנבחר הוא $SARIMA(3,0,5)(1,0,2)$. ניתן לראות כי בעת נבחר מודל יותר מורכב עם יותר פרמטרים. הצלחנו להציג זאת על ידי מתן רמזים נכונים לפונקציה שיכלה לבצע את החיפוש תוך שימוש בקבוצת פרמטרים מצומצמת הרלוונטיות לפתרונו הבעיה שלנו.
- ציון AIC של המודל היה 365.245. ניתן לראות כי הצלחנו לקבל ציון AIC נמוך משמעותית מהציון שקיבלו בעבר ההערכתה הקודמת (וכעת התוצאות כן ברוח השוואה!).

- זמן הריצה של הפונקציה היה כעת ארוך משמעותית, מכיוון שביקשו מהפונקציה לסרוק את כל היצירופים האפשריים עבור הפרמטרים השונים בתוך גבולות המרחב שיצרנו.

נרצה לבדוק האם אכן ישנו שינוי בין החיזוי של שני המודלים השונים שקיבלנו עבור כל אחת מההרצאות. לשם כך ננסה לחזות את אותה תקופה הזמן בעזרת כל אחד מהמודלים, ונשווה בין ביצועי המודלים. לצורך הניסוי נחoze תקופה זמן של כ-10 ימים החל מהתאריך 20/07/2021. נציג את תוצאות החיזוי על גרף :



מבחן גרפית ניתן לראות כי מודל SARIMA מתאים את החיזויים שלו בצורה טובת יותר ממודל ARIMA הנאיבי.

נציג את המטריקות עבור כל אחד מהמודלים :

```
####--Metrics for model accuracy---###
Test Data Mean: 1770.8
Mean Absolute Error 585.3063355313177
Mean Squared Error: 491663.0049335426
Mean Absolute Percentage Error: 0.2964328980358729
Root Mean Squared Error: 701.1868545070869
Median Absolute Error: 565.9571635319005
R2 score: -0.9820105401292121
```

עבור : $ARIMA(1,0,3)$

אחו השגיאה הממוצع עומד על כ- 0.29 , וציון $R2 score$ שלילי מצביעים על דיקוק לא טוב של המודל.

עבור 7 : $SARIMA(3,0,5)(1,0,2)$

```
#####-Metrics for model accuracy---#####
Test Data Mean: 1770.8
Mean Absolute Error 108.78049222055174
Mean Squared Error: 21936.581440979786
Mean Absolute Percentage Error: 0.07552851553571029
Root Mean Squared Error: 148.1100315339234
Median Absolute Error: 62.71684570230275
R2 score: 0.9115684214713252
```

אחוֹ השגיאה הממוצע עומד על 0.07 וציוֹן $R2 score$ עומד על 0.91 . תוחלת השגיאה עומדת על כ- 108.78 וחציוֹן השגיאה עומד על 62.71 , כל זאת בהשוואה לתוכלת ערכיוֹ ה- $test set$ העומדת על כ- 1770.8 .

ניתן לראות במקורה זה בבירור את היתרון של המודל המורכב יותר שהתקבל מושכלת של פונקציית auto arima .

Rolling Forecast

בחלק זה נרצה להציג בעיה שקיים לנו כאשר אנחנו מנסים לחזות תקופות זמן ארוכות קדימה, ולהציג פתרון לבעיה. כדי להמחיש את הבעיה נאמר שאנו רוצים לחזות את מס' המאומתים היומיים בrama הלאומית, ולספק חיזוי של חודש ימים, החל מהתאריך $01/02/2021$ ועד לתאריך $01/03/2021$. עבור עבודה החיזוי בחלק זה נבחר להשתמש במודל ARIMA. זהו מודל מקיף המכיל בתוכו את כל האלמנטים של time series שארינו בחצגנו בפרק הקודם (MA ו-AR), ובנוסח בעל יכולת לטיפול בסיטציוני בעזרת החלק h-integrated במודל. יתרון נוסף של המודל הוא שזמן האימון שלו מהיר יותר בהשוואה למודל SARIMA. מכיוון שבחלק זה נבצע אימונים רבים של המודל (כפי שנראה בהמשך), יש יתרון בבחירה של מודל פשוט יותר וקל לאימון.

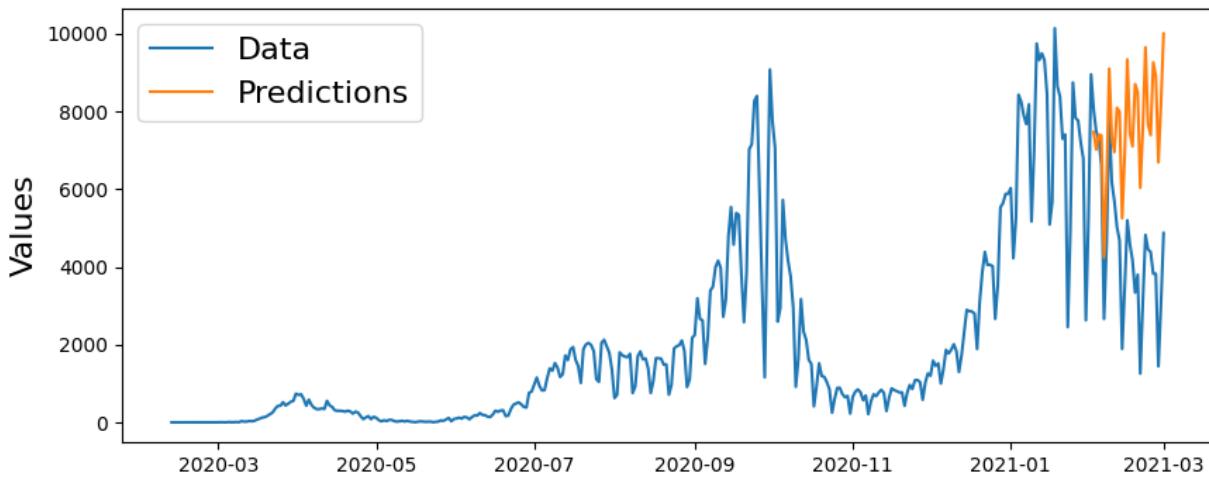
בפרק הקודם ביצענו את תהליך בחירת הפרמטרים עבור המודל, וראינו כי $ARIMA(7,2,7)$ מלהווה בחירה טובה של פרמטרים עבור המודל. מכיוון שחלק זה לא עוסק בכוונון פרמטרים, נסתפק בשימוש בפרמטרים הללו עבור ההרצאות של המודל.

ננסה אם כך לחזות את תקופת הזמן המדוברת בעזרת שימוש במודל $ARIMA(7,2,7)$. נציג כפלט את גراف השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:

Residuals from ARIMA Model



Data vs. Predictions



נציג את המטריקות השונות עבור דיקוק המודל:

```
##### Metrics for model accuracy #####
Test Data Mean: 4560.857142857143
Mean Absolute Error 3154.2255470509785
Mean Squared Error: 12874866.287647372
Mean Absolute Percentage Error: 0.9857010577665973
Root Mean Squared Error: 3588.1563911913554
Median Absolute Error: 3269.1394913545737
R2 score: -3.089559055547512
```

נרצה תחילה לנתח ולהבין את התוצאות שקיבלנו. ניתן לראות מהגרפים וממטריקות המדידה של המודל שהחיזוי שלנו מציג סטיות נדולות מאד ורחוק מאד מערכי האמת. בפרט ניתן לראות מגף ה-*residuals* כי השגיאה בחיזוי גדלה ככל שאנו מתקדמים בתקופה אותה אנחנו חוזים. נזכיר כי תוצאות אלו מתקבלות למרות שהמודל שבחרנו הוא מודל שנבחר בצורה טובה עם פרמטרים שהותאמו בצורה נכונה לנ נתונים שלנו.
אם כך, מדוע אנחנו מקבלים שגיאה כל כך גדולה בתוצאות החיזוי?

הבעיה כאן טמונה בתקופת הזמן אותה אנחנו חוזים. כאשר אנחנו מנסים לחזות חדש שלם קדימה, הערכים הראשונים אותם נחזה יהיו מבוססים על ערכי האימון, אך ככל שנתקדם אל תוך התקופה אותה אנחנו חוזים, הערכים שנחזה לא ייקבעו על סמך ערכי האמת של התוכנה, אלא על סמך ערכי החיזוי שבו פניהם. בצורה זו, ברגע שישנה שגיאה בערכי החיזוי, עצמת השגיאה מתעצמת ככל שאנו מתקדמים בזמן, וערכים החיזוי עלולים ליצור גורף חיזוי השונה לחלוthin מגף ערכי האמת (בדיקות כפי שקרה לנו). תופעה זו גם מסבירה את הדפוס שקיבלנו בגרף ה-*residuals*.

ניתן להסתכל על הבעיה בצורה יותר אינטואיטיבית – בהינתן שנתקבקש לחזות את העתיד בנוגע לתוכנה כלשהי, יהיה הרבה יותר קל לחזות את העתיד הקרוב (יום או יומיים קדימה) לעומת חיזוי ישיר של מה שיקרה עוד 30 ימים, ללא שום ידע על 29 הימים האחרים אשר מפרידים ביןינו לבין היום אותו אנחנו רוצים לחזות.

כדי לפטור את הבעיה, נניח הנחה חשובה אשר לרוב מקובל להניח בעת פתרון בעיות time series – כאשר אנחנו באים לחזות את יום t , ניתן להניח כי כל ערכי האמת של התוכנה אותה אנחנו חוזים ידועים לנו עד

ליום $t - 1$. באמצעות ההנחה (המעט מסקלה) זו, נוכל כעת להציג אלגוריתם לפתרון הבעיה שהציגנו לעיל. על מנת לחזות תקופה של חודש ימים, נחזה תקופה יומ בודד, ולאחר מכן נשתמש בערך האמת של התוכנה עבור היום שחזינו כדי לאמן את המודל שלנו בשנית. לאחר מכן נחזה את היום הבא בהתבסס על האימון החדש שביצענו, וכך הלאה.

פואדו קוד עבור אלגוריתם **rolling forecast**:

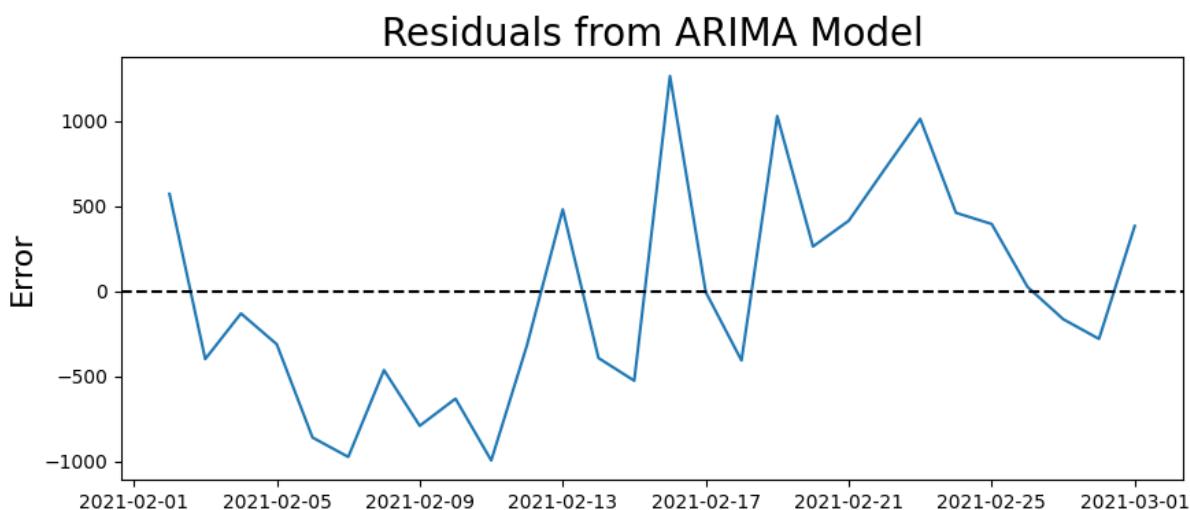
```

1 initialize rolling_predictions
2 for end_date in test_set:
3     train_data = time_series up to end_date
4     fit_model(train_data)
5     forecast 1 day ahead
6     append prediction to rolling_predictions
7 return rolling_predictions
8

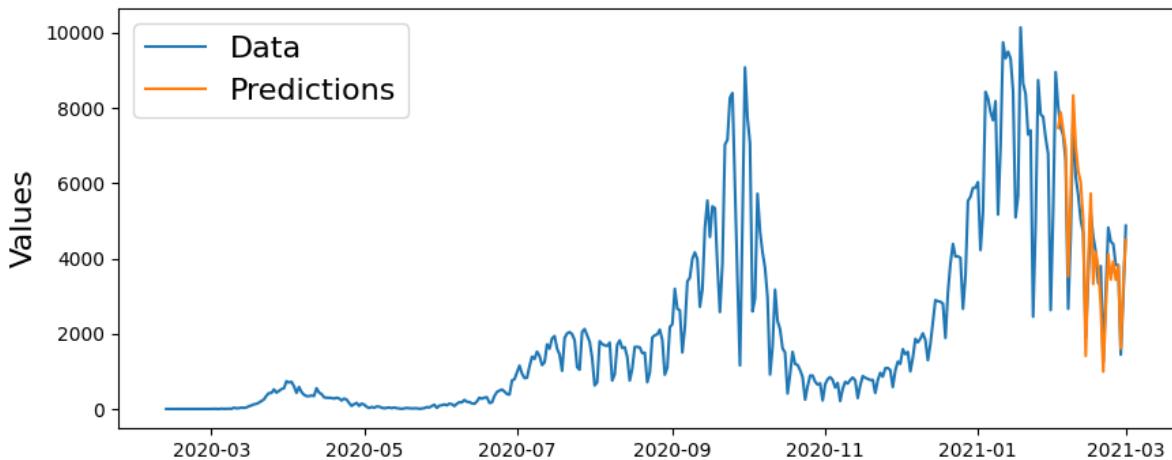
```

בעזרת אלגוריתם זה נוכל למעשה לגשר על הפער שנוצר לנו מה צורך לחזות כל כך הרבה ערכים קדימה, ובתקופה לשפר בכך את איכות החיזוי. אלגוריתם זה יכול להיות שימושי במיוחד כאשר אנחנו מנסים לחזות תקופה בה ישנו שינויים במגמה הכללית של הנתונים (נניח תקופה בה פונקציית trend משנה מגמה מעלה לירידה או להפך). החיסרונו בשימוש באלגוריתם זה הוא שהוא מבוסס על אימון חדש של המודל בכל יום (retrain), וכן למעשה נוכל לספק תחזית רק עבור יום אחד קדימה בכל פעם.

נסה כעת לחזות את אותה תקופה הזמן בעזרת המודל ARIMA(7,2,7) תוך שימוש באלגוריתם **forecast** עבור החיזוי. נציג כפלט את גרפ' השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:



Data vs. Predictions



```
##### Metrics for model accuracy #####
Test Data Mean: 4560.857142857143
Mean Absolute Error 524.0102201731133
Mean Squared Error: 375254.36580285715
Mean Absolute Percentage Error: 0.13008551994871598
Root Mean Squared Error: 612.5800892967851
Median Absolute Error: 436.17711667686444
R2 score: 0.8808045959067404
```

מציג את המטריקות השונות עבור דיקון המודל:

ניתן לראות כי המגמה השלילית של גף residuals נעלמה וcutout השגיאות נעות סביבה 0 לכל אורך תקופת החיזוי. בנוסף לפיקוח מטריקות הדיקון ניתן לראות כי קיבלנו אחוז שגיאה 0.13 וציון R^2 של 0.88. כל אלו מהווים שיפור דרמטי בהשוואה לחיזוי המקורי שביצענו ללא שימוש באlgorigitms .rolling forecast.

המטרה העיקרית בחלק זה הייתה להראות את החשיבות הגדולה של אימון מחדש (retrain) של המודל שלנו. למרות שבשני המקרים המודל התאמן על תקופה של כמעט שנה, ההבדל העיקרי ביביצועים הגיע בזכות האימון מחדש של המודל שהשתמש ב-rolling forecast.

הרחבה לאלגוריתם – החישורו המשמעותי באlgorigitms rolling forecast בגרסתו הבסיסית היא שהוא מבצע אימון מחדש עבור כל יום. המשמעות היא שנוכל בפועל לחזות רק יום אחד קדימה. כדי להסיר את המגבלה זו נציג הרחבה אפשרית לאlgorigitms שתיקרא *rolling_forecast(k)*. הפרמטר *k* יקבע את מס' הימים אותם נזהה בכל איטרציה לפני ביצוע האימון מחדש.

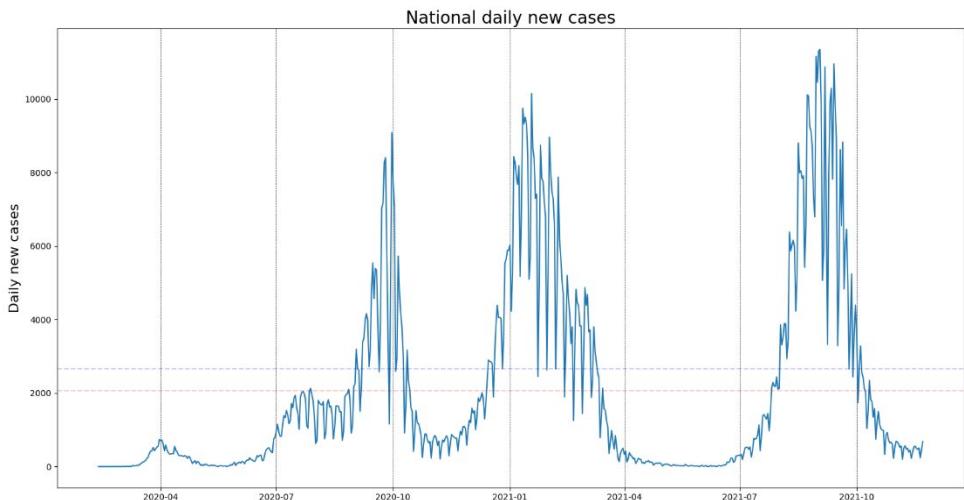
פואדו קוד עבור אלגוריתם *rolling_forecast(k)* (לשם הנוחות נניח כי *k* הוא קבוע של גודל

```
1 initialize rolling_predictions : (test_set-
2 end_date = train_set.end_date
3 while end_date <= test_set.end_date:
4     train_data = time_series up to end_date
5     fit_model(train_data)
6     forecast k days ahead
7     append predictions to rolling_predictions
8     end_date += k
9 return rolling_predictions
```

קיים tradeoff בבחירה *k* – ככל שנבחר את הפרמטר *k* להיות גדול יותר כך נוכל לחזות יותר ימים קדימה לפני כל *retrain*, אך תוצאות החיזוי צפויות להיות פחות טובות.

Rolling Average Smoothing

نبיט שוב על ה-time series שלנו המציג את מס' המאומתים הימיים בrama הלאומית:



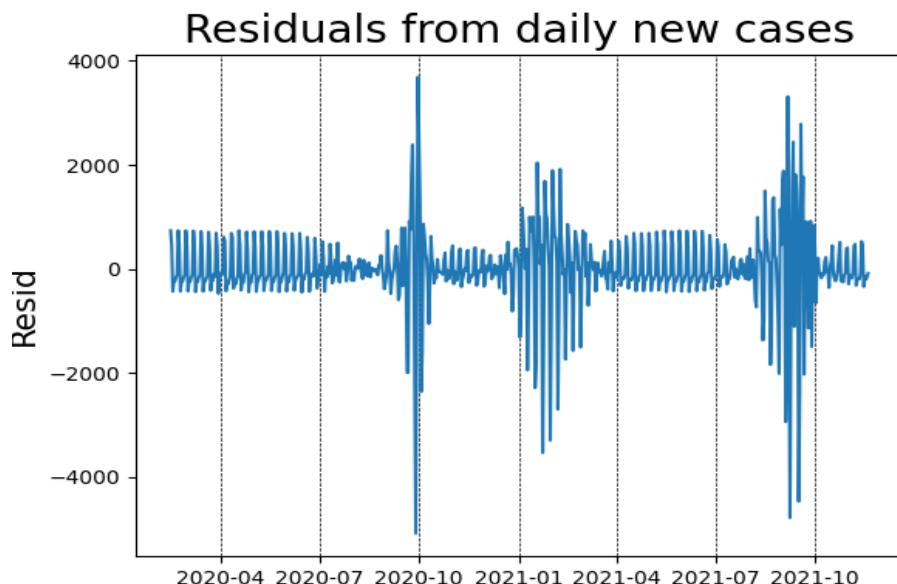
שתי הבעיות המרכזיות הקשורות על חיזוי מדויק של התוכונה שלנו הן :

1. וולטיליות גבוהה (volatility) - בקונטקטנו בו אנחנו עוסקים, משתמש במונה זה על מנת לתאר אי יציבות בסטיית התקן של הדגימות לאורך זמן. מבחינה אינטואטיבית הכוונה היא שהגרף המתאר את ה-time series שלנו לא "חלק".
2. עונתיות שבועית (כפי שהסבירנו בחלקים הקודמים).

ניתן לראות כיצד שני המאפיינים הללו באים לידי ביטוי בגרף שהוצג לעיל. בפרק זה ובפרקים הקודמים ה证实נו רבות עם שתי הבעיות הנ"ל. הצענו דרכי רבות להתחמזר עם הבעיות שקיימות לנו בתנאים :

- הפעלת טרנספורמציות על ה-data להגברת מידת הסטציונריות.
- מודלים המתחשבים בעונתיות (SARIMA).
- כיוונו פרמטרים (auto arima).
- אלגוריתם rolling forecast

כל הכלים בהם השתמשנו אכן הצליחו להביא את המודלים שלנו לbijouxים יפים מאד עבור תקופות חיזוי מסוימות, אך עדין לצערנו יהיו מקרים בהם מודלים אלו ייכשלו לחזות בצורה אמינה, וזאת בשל כמויות גדולות של "רעש" ואי יציבות שלא ניתן לחיזוי. ניתן לראות בmphorsh את מידת הוולטיליות של ה-time series שלנו בעזרת שימוש בפונקציית `seasonal decompose` בה השתמשנו עבור מודל ה-SARIMA ובבחינה של גראף ה-residuals של הפונקציה :



כזכור, גраф זה מציג את השגיאות שלא נכללות כחלק מגраф trend או seasonality, ומיחסות לרוב השגיאות של "רעש לבן". במקרה שלנו ניתן לראות כי תהליכי השגיאה הוא מאד משמעוני וערכיו יכולים לנوع מ- -4000 ועד +4000+. ניתוח זה מביא אותנו למסקנה כי ניסיון להתחקות אחר תהליכי שגיאה שכזו עלול להיות מאד קשה.

על מנת לפטור את הבעיה אנחנו מבינים כי علينا לבצע שינוי או טרנספורמציה כלשהי לבעיה אותה אנחנו חוזים, במטרה להקטין את רמות אי היציבות, "להחליק" את הנתונים שלנו ובתקווה גם להיפטר מהעונתיות שקיים לנו ב-time series. כדי שימושי עבור מצבים כאלה הוא שימוש בממוצע נع כתחליף לערך התוכונה. המשמעות היא שבמוקום להסתכל על ערך התוכונה עבור יום כלשהו, ערך התוכונה עבור אותו היום יהיה ערך הממוצע של התוכונה עבור מס' ימים כלשהו של ימים (כולל היום אותו אנחנו בוחנים). סוג כזה של ממוצע נקרא ממוצע נע (rolling average). באופן לא מפתיע, בחירה טבעיות עבור הבעיה שלנו תהיה של ממוצע נע שבועי (7 day rolling average). הבחירה בשבועו מצד אחד מספק גודלה בכך לעוזר לנו "להחליק" את הגראף, ומצד שני היא לא גודלה מדי ובכך לא מנוגנת את הנתונים ושומרת על trend הדומה לזה של הבעיה המקורי שלנו. בנוסף, לאור העובדה כי מס' הבדיקות בסופי שבוע נמוך מימי חול, הבחירה בממוצע נע שבועי תעוזר לכסות את הפערים הללו בצורה טובה.

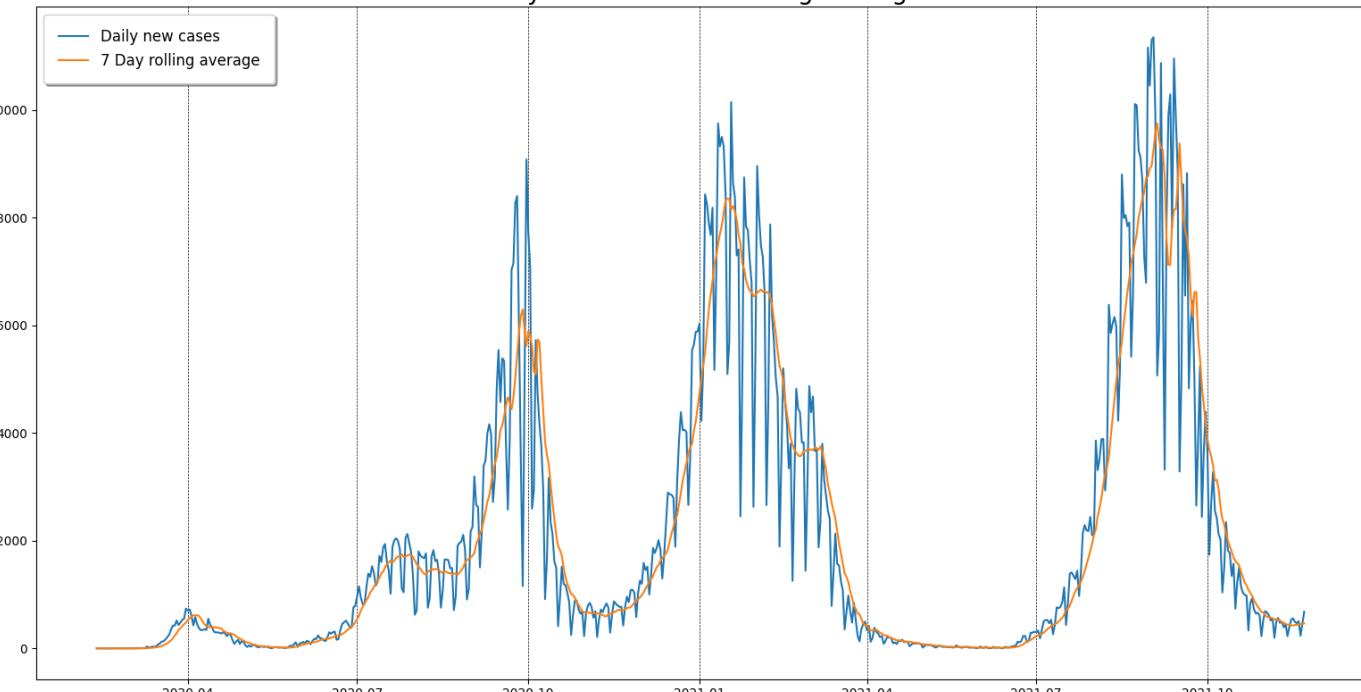


הבחירה במעבר מהבעיה המקורי שלנו לבעיה הממוצע הנע אינה מקרית. זהهي פרקטיקה מקובלת בפתרונות בעיות time series ככלל ובניתוח סטטיסטי של תחלואה בפרט. ניתן לראות כי בלוח הבקרה של משרד הבריאות משתמשים גם כן בממוצע הנע השבועי על מנת להציג את מגמות התחלואה:

חשיבות להציג כי בעת אנחנו מתמודדים עם בעיתת חיזוי חדשה. המשמעות היא שלא יהיה נכון מבחינה תיאורטית להשוו את התוצאות שנקבל עבור בעיתת הממוצע הנע לתוצאות שקיבלנו עבור חיזוי מס' המאותיים היומי. עם זאת, מטרתנו היא לשכנע כי הביעות הללו הן מספיק דומות, וכי חיזוי טוב של הביעת החדשה יוכל לעזור לנו לחזות טוב יותר גם את הביעת המקורית שלנו.

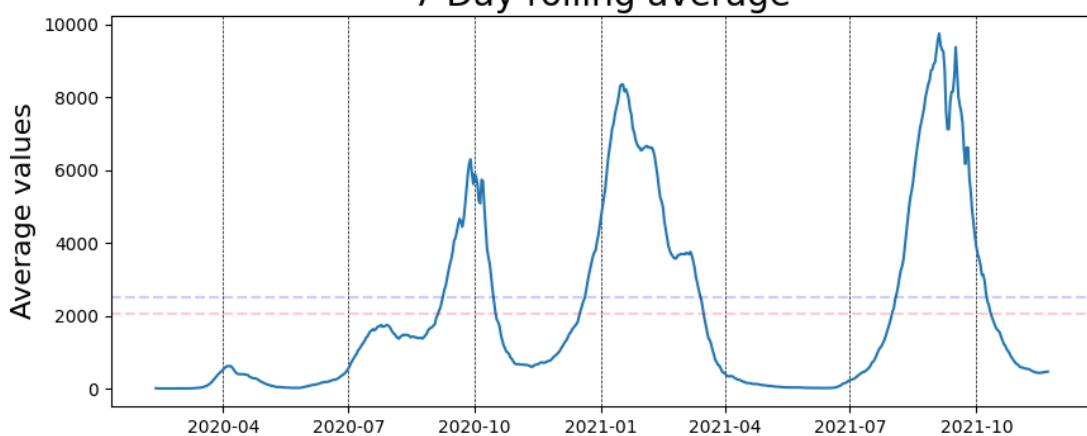
כדי להדגים את הקשר בין גרפ' הממוצע הנע לבין הביעת המקורית שלנו, נראה את ההתנהגות של שתי התכונות בגרף אחד:

Daily new cases vs. Rolling average



ראשית, ניתן לראות כי המוגמות הכלליות של שני הגרפים דומות. בנוסף ניתן לבחין שעבור הממוצע הנע יש לנו הרבה פחות אי יציבות והגרף אכן "חלק" יותר, כפי שקייםנו. נציג כעת רק את הגרף עבור הממוצע הנע:

7 Day rolling average



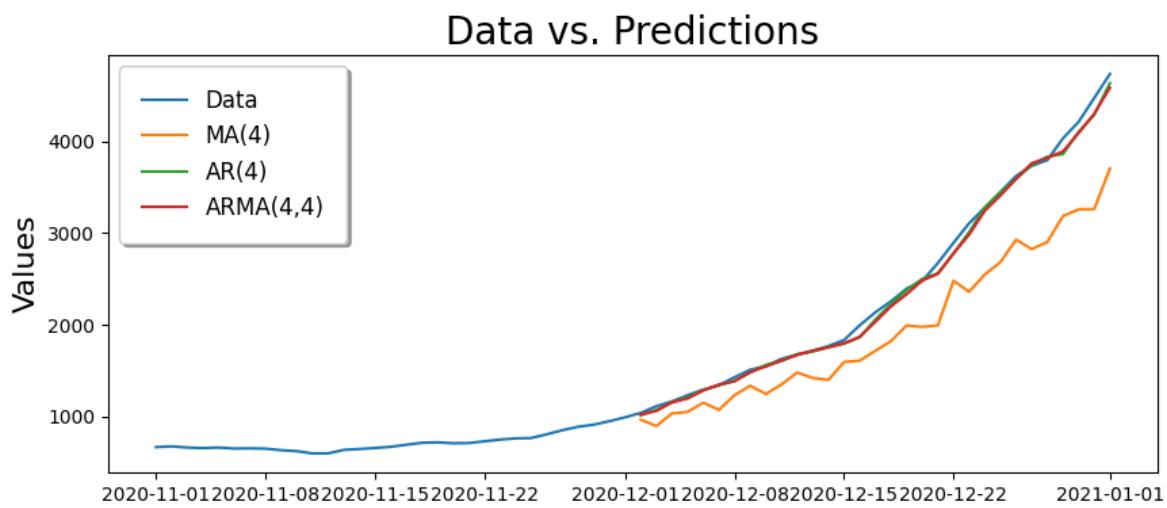
כפי שניתן לראות כתם הצלחנו להיפטר מהעונתיות שהייתה קיימת לנו בבעיה המקורית, ונשארנו עם גרפּ trend חלק. גרפּ זה הוא גרפּ משמעותי יותר קל לחיזוי, כפי שנראה מייד.

```
Augmented Dickey-Fuller Test Results:
ADF Test Statistic      -3.903294
P-Value                  0.002011
# Lags Used              18.000000
# Observations Used     631.000000
Critical Value (1%)      -3.440756
Critical Value (5%)      -2.866131
Critical Value (10%)     -2.569215
dtype: float64
Is the time series stationary? True
```

תחילה נרץ ADF test כדי לבדוק את מידת הסטציאונריות של ה-data :
קיבלו Ci ה-data שלו סטציאוני, וזה לא שהינו צריכים להפעיל טרנספורמציות כלל.

נוכל אם כך לגשת יישורות לתהיליך החיזוי. נראה תחילת השוואה בין המודלים השונים בפרק הקודם על ה-time series החדש שלנו. לאחר מכן נבצע חיזויים עבור גל התפרצויות השלישי והרביעי. גם במקרה זה, בדומה לחלק הקודם, לא נתמקד בכוונו הפרמטרים הטובים ביותר עבור כל מודל בו אנחנו משתמשים, וננסה להשתמש במודלים הפשטוטים ביותר אשר ייתנו לנו תוצאות חיזוי טובות (תוך זמן אימון סביר של rolling model). מכיוון שאנו רוצחים לחזות תקופות זמן של מס' חודשים בכל פעם, נשתמש באלגוריתם forecast שהוזג בחלק הקודם.

לצורך הדוגמה הראשונה נשתמש בשלושת המודלים הבסיסיים אותם הצגנו בחלק הקודם – MA, AR ו- ARMA. הפרמטרים עבור המודלים נבחרו בהסתמך על פונקציות ה-ACF ו-PACF. כאמור, החיזוי יבוצע באמצעות שימוש באלגוריתם rolling forecast (עם *retrain* לאחר כל יום). נבצע חיזוי של חדש בין התאריכים 1/12/2020 – 1/1/2021 – זהו תאריך סוף תקופת האימון) ועד ה-1/1/2021 (כולל). נציג את תוצאות החיזוי בגרף. טווח התאריכים המוצג בגרף מצומצם כדי להציג את ההבדלים בין המודלים השונים :



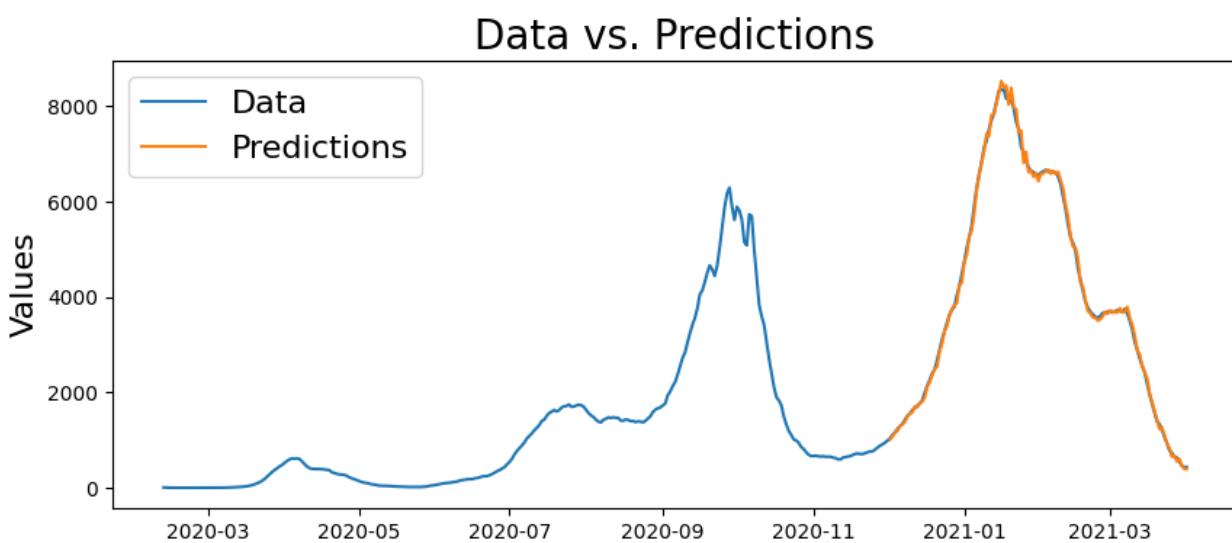
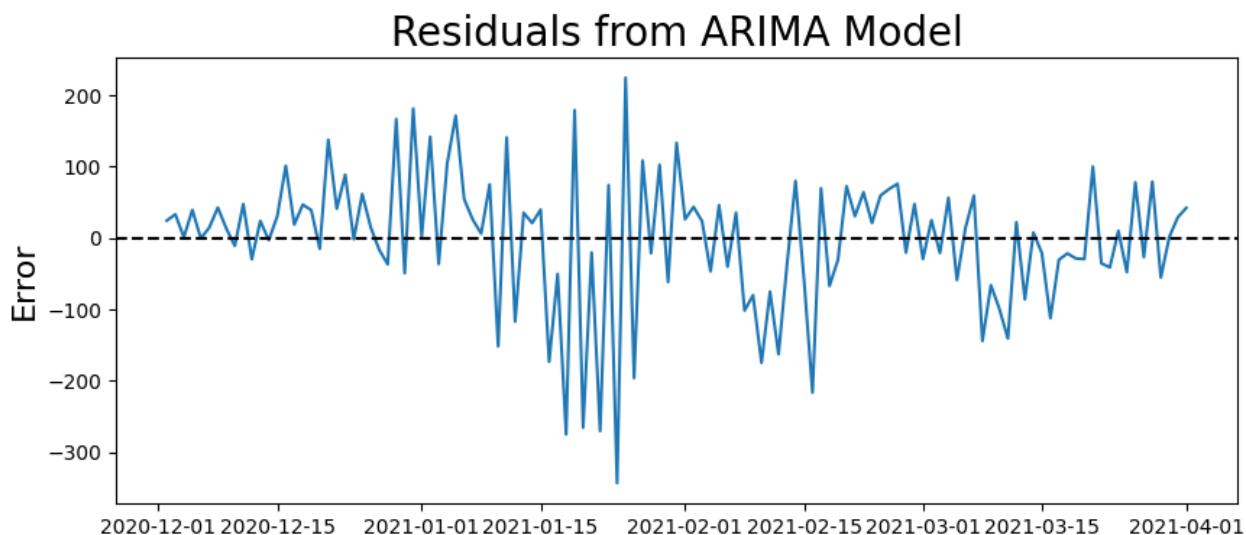
מציג את המטריקות הנבחרות עבור כל מודל בטבלה :

	MA	AR	ARMA
<i>R2 score</i>	0.72	0.995	0.995
<i>MAPE</i>	0.18	0.019	0.021

ניתן לראות שיפור דרמטי בביצועים של האלגוריתמים שלנו בזכות המעבר ל-time series המוחלך ובזוכות השימוש ב-*rolling forecast*.

חיזוי גל ההתרצות השלישי – בין התאריכים 01/12/2020 ו-01/04/2021 ועד ל-01/04/2021

ננסה כעת לחזות את תקופת ההתרצות של גל התחלואה השלישי באמצעות המודל $ARIMA(2,1,2)$ תוך שימוש באלגוריתם *rolling forecast* עבור החיזוי. נציג כפלט את גרע השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת:



מציג את המטריקות השונות עבור דיקט המודל :

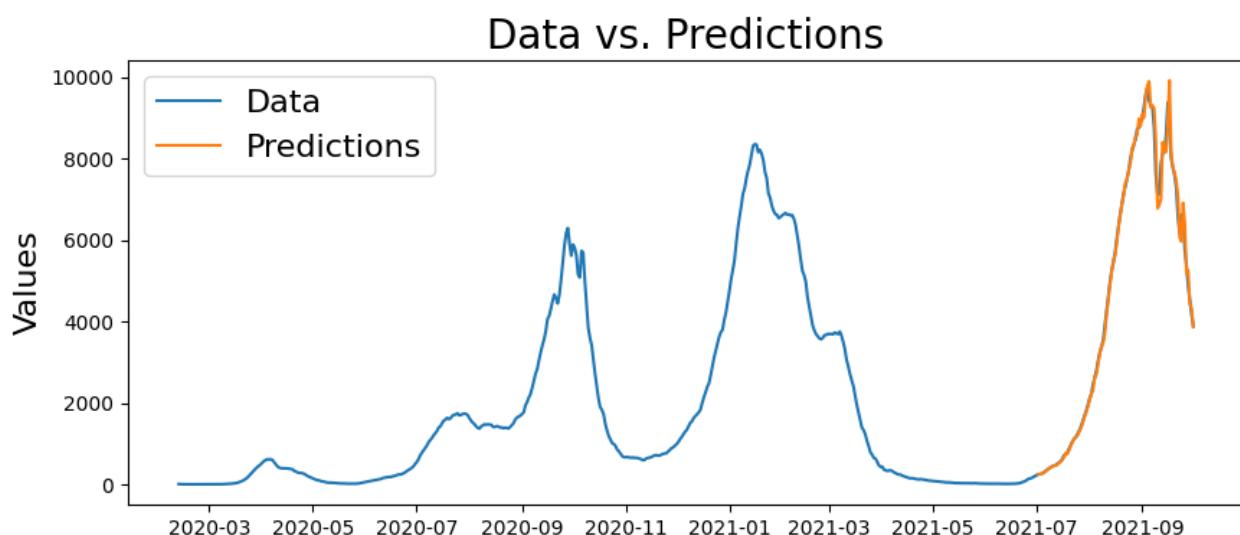
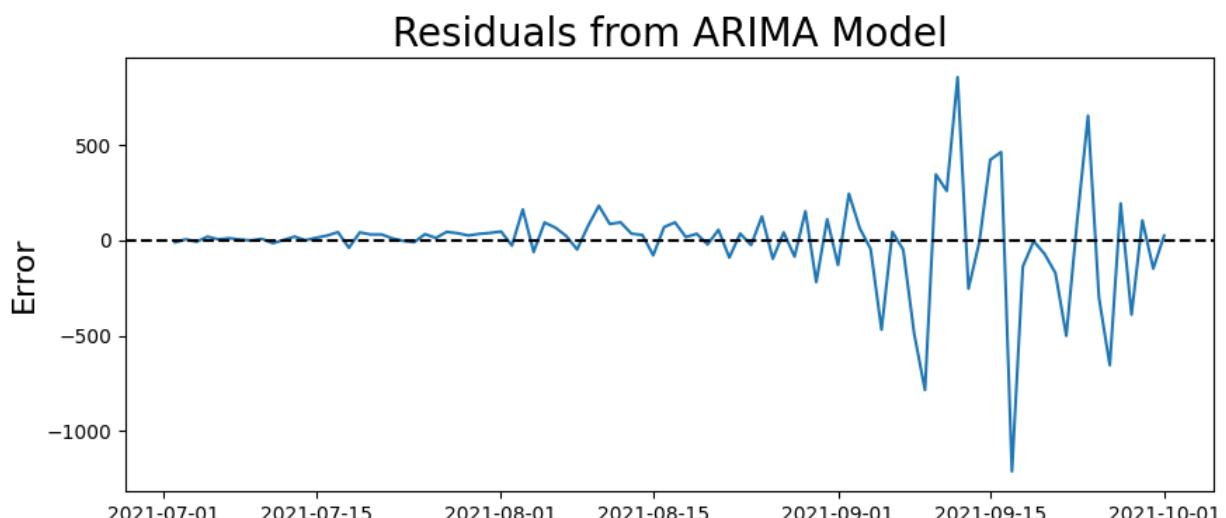
```
#####Metrics for model accuracy#####
Test Data Mean: 4127.330578512397
Mean Absolute Error 69.84746537340305
Mean Squared Error: 9125.781212513482
Mean Absolute Percentage Error: 0.021301986617236625
Root Mean Squared Error: 95.52895483838124
Median Absolute Error: 46.84982140228203
R2 score: 0.9983416438025825
```

אחוז השגיאה שקיבלו עומד על כ-0.02 בלבד ! תוחלת השגיאה עומדת על כ-69.8 בהשוואה לתוחלת ערכי test set העומדת על כ-3.4127.3. בנוסף, ציון ה-*R2 score* עומד על 0.998. אלו תוצאות מאד מרשימות.

יש לזכור כי השימוש ב-*rolling forecast* למעשה אומר כי אנחנו מבצעים בכל פעם חיזוי של ערך בודד, ולבסוף אוספים את כל הערכים ומציגים אותם בתור פلت החיזוי המלא.

חיזוי גל ההתרצות הרביעי – בין התאריכים 01/07/2021 ו-01/10/2021 ועד ל-01/10/2021

ננסהikut לחזות את תקופת ההתרצות של גל התחלואה הרביעי בעזרת המודל *ARIMA(2,1,2)* תוך שימוש באלגוריתם *rolling forecast* עבור החיזוי. מציג כפלט את גרפ' השגיאות (residuals) ואת ההשוואה בין ערכי החיזוי לערכי האמת :



```
#####-Metrics for model accuracy---###
Test Data Mean: 4687.119565217391
Mean Absolute Error 136.63836064999552
Mean Squared Error: 62599.35608272219
Mean Absolute Percentage Error: 0.02837147102914866
Root Mean Squared Error: 250.1986332551043
Median Absolute Error: 47.17304117392314
R2 score: 0.9940427666954874
```

נציג את המטריקות השונות עבור דיקט המודל:

אחוז השגיאה שקיבלנו עומד על כ-0.02 בלבד! תוחלת השגיאה עומדת על כ-136.6 בהשוואה לתוחלת ערכי ה-test set העומדת על כ-4687.1. בנוסף, ציון ה-*R2 score* עומד על 0.994. גם עבור גל ההתפרצויות הרביעי קיבלנו תוצאות חיזוי מעולות.

סיכום

בחלק זה הצגנו שלושה ניסויים מרכזיים שונים, כאשר בכל ניסוי ניסינו לגשת לביעית החיזוי שלנו מכיוון שונה, ולהציג פתרונות שונים ומגוונים לביעות איתן נתקלנו בעת העבודה עם הנתונים. הצגנו את הגישה הקלאליסטית של כוונון הפרמטרים עבור האלגוריתם שלנו, והראינו כיצד מעט ידע חיצוני מאפשר לנו לעוזר כלים אוטומטיים לכוונן את הפרמטרים בצורה ישרה יותר ובקבות כך גם לתוצאות חיזוי יותר טובות. לאחר מכן הצגנו את אלגוריתם ה-rolling forecast אשר אפשר לנו לבצע חיזוי מדויק של תקופות זמן ארוכות. השיפור המשמעותי שהושג באמצעות השימוש באlgorigthms ה-rolling forecast מדגיש את החשיבות באימון חוזר (retrain) של המודל שלנו. לבסוף הראינו כיצד מעבר מהבעיה המקורית שלנו לביעה דומה אחרת (בעיית ה-e-average) יכול להביא לשיפור דרמטי בדיקת המודלים שלנו על ידי כך שאנחנו מנטרלים את הביעות שהוא לנו ב-time series המקורי שלנו, לרבות וולטיליות, עונתיות וחוסר בסטציונריות.

סיכום הפרויקט

בפרויקט זה ניסינו לפתרו את בעיית חיזוי התחלואה בקורונה בישראל. מטרתנו הייתה להראות כי ניתן לגשת לבעיה ולפתרו באמצעות תוקן שימוש במספר מודלים ואלגוריתמים שונים. תחילה ניסינו לפתרו את הבעיה בעזרת שימוש באלגוריתמי סיווג קלאסיים. הקושי המרכזי איתנו היו צריכים להתמודד בחלק זה היה התאמה של בעיית החיזוי שלנו השicket המשפחתי בעיות time-series, לאלגוריתמי סיווג כגון עצי למידה ו-KNN. בחלק זה התבמכו הרבה באיסוף של מידע ובנית תוכנות אשר יסייעו לנו ביצוע החיזוי הנדרש, כמו כן ביצעו את ההתאמות הנדרשות עבור אלגוריתמי הלמידה הקלסיים על מנת לקבל את החיזוי הטוב ביותר. לאחר מכן, עברנו לפתרון הבעיה בצורה "טבעית" יותר באמצעות אלגוריתמי time-series. הצגנו משפחה של אלגוריתמי למידה המבוססים על רגרסיה לינארית, והראינו כיצד ניתןציה בצורה נכונה את家庭 time-series שלנו, להפעיל עליו טרנספורמציות ולבצע בחירת פתרונות מתאימים עבור אלגוריתמי החיזוי השונים.

דיון בתוצאות

אלגוריתמי למידה קלאסיים

- עצי החלטה – נעשה שימוש נרחב בשני מודלים כאשר האחד מייצג עץ החלטה בודד ואילו השני יער החלטה המכיל מספר עצי החלטה הנגנים מדוגמאות שונות מתוך set-train. בשימוש במודלים אלה ניתן דגש גדול לתוכנות time-set-training אליו עבדנו. לאורך הדרך ראיינו כי בחירת תוכנות חכמה מניבה תוצאות מדויקות יותר וזאת כיון שקיימות תוכנות רלוונטיות יותר ורלוונטיות פחותה בבנייה המודלים. בנוסף, נעשתה פעולה נוספת אשר באה להתמודד עם "התאמת יתר" ובכך הצלחנו לשפר אף יותר את התוצאות אשר התקבלו לאחר בחירת התוכנות הטובות ביותר. לאור האמור לעיל נסיק כי תוכנות אלה ממחישות באופן מובהק כי השימוש מקדים של set-train בצורהמושכלת ופועלות נוספת בתחום העצם משפרים את אמינות המודלים ובכך עוזרים לנו לקבל תוצאות טובות יותר.

- KNN – השימוש באלגוריתם KNN הניב תוצאות יפות, וזאת למורות שלא מדובר באלגוריתם ייעודי לפתרון בעיות time-series. הצלחנו בעזרת הכלים שתיארנו בחלק ה-KNN לשפר את הדיקט בזורה ניכרת שעלה על ציפיותינו. גילינו שעליינו להיות זהירים מאוד כאשר משתמשים באלגוריתם KNN לפתרון בעיות time-series מכיוון שבחירה set-train לא יכולה לחפות עם בחירת set-test כדי לדמות שימוש אמיתי. בנוסף, גילינו שהאלגוריתם (ומטריקות המדידה בהן משתמש האלגוריתם) הגיע מאד למומצע ערכי set-test, כאשר ממוצע זה שואף ל-0 או מטריקות הדיקט מניבו תוצאות שמאבדות משמעות (לא בין 0 ל 1). מושם כך בחרנו את set-test כך שממוצע הערכים שלו לא שואף ל-0 וקיבלו תוצאות הגיונית שעל בסיסן ביצעו את האימון (כיווננו הפתמריטים). בהתאם לנקודת הקודמת, כאשר בדקנו את דיקט האלגוריתם על מאפיינים ספציפיים set-test-mean, גילינו שככל שה-mean גבולה יותר, כך הדיקט שלנו גבוה יותר. למרות זאת, כאשר הממוצע נמוך, אומנם הדיקט נמוך יותר אך במספרים מוחלטים השגיאה יחסית קטנה, ולכן בעיית חיזוי הקורונה, לדעתנו התוצאה מספקת.

אלגוריתמי time-series

בחלק זה התבמכו בחיזוי התחלואה ברמה הלאומית. ראיינו כי גרפ הפונקציה אותה אנחנו מנסים לחזות הוא גרפ קשה לחיזוי, מכיוון שהוא לא סטציוני, ולטילי (volatile) ומכיל מרכיב של עונתיות. על מנת

להתגבר על בעיות הסטציונריות, בינו טרנספורמציות אשר עזרו לנו להביא את הנתונים למכב מעת יותר יציב. לאחר מכן התחלנו במלאת החיזוי. תחילת השתמשנו באלגוריתמי הלמידה הבסיסיים MA ו-AR, אשר כל אחד מהם משתמש בפרדיקט לינארי אחד על מנת לבנות את משוואת החיזוי של המודל. אלגוריתמים אלו התקשו בחיזוי ונתנו תוצאות לא אופטימליות. ההשערה שלנו היא שהקשיי בחיזוי בעורת מודלים אלו נבע מהਮורכבות של time series שלנו ולעובדה כי אלו מודלים פשוטים אשר מסתמכים על פרדיקט בלבד, ובכך לא מצליחים להעריך בצורה טוביה יותר התנוגויות מורכבות (כגון עונתיות לדוגמה). כבר בשימוש במודל ARMA, אשר משלב את שני המודלים MA ו-AR, רأינו שיפור משמעותית בתוצאות החיזוי. תוצאה זו מASHת את ההשערה שלנו שהוספה של פרדיקטים לשמש את המודל מחזקת את יכולות החיזוי שלו. על מנת לטפל בבעיות העונתיות, השתמשנו במודל SARIMA אשר שילב גרסיה לינארית המבוססת על ערכי פרדיקטים קודמים, בשלוב עם גרסיה עונתית אשר לוקחת בחשבון השפעה של תקופות עונתיות קודמות על היום הנוכחי אותו אנחנו מנסים לחזות. השימוש במודל זה הצליח להביא לתוצאות חיזוי מרשימות למדוי, ובכך אישש את הצורך בהתחמಡות טובה עם העונתיות בתנאים שלנו על מנת להצליח להגיע לאחוזי דיקוגניטיביים בחיזוי.

בחלק האחרון בפרק זה, ביצענו טרנספורמציה לבעית החיזוי המקורי שלנו (חיזוי מס' המאומתים היומיי ברמה הלאומית) לבעית חיזוי הממוצע הנع (rolling average). ביצוע הטרנספורמציה לבעית החיזוי החדש חשב לפניו גרפ חדש שבו נעלמו כל "הבעיות" שהיו לנו בגרף הפונקציה המקורית – קיבלו גרפ חליך, ללא עונתיות ובעל תוכנות סטציונריות. כתוצאה לכך, נוכחנו לגלוות כי החיזוי על גרפ הפונקציה החדשה הפך להיות משמעותית יותר קל. יתרה מכך, רأינו שגם באמצעות מודלים פשוטים יותר (וללא שימוש בכוננו פרמטרים) עדין הצלחנו להשיג תוצאות חיזוי מרשימות יותר בהשוואה לבעיה המקורי (חיזוי מס' המאומתים היומיי). המשקנה העיקרית שלנו מחלוקת זה הייתה חישיבות גדולה מאוד לתוכנות הסטטיסטיות של time series שלנו על היכולת להפיק חיזוי מדויק, וכי לעיתים עדיף להשקיע מאמץ בעבודה על התנאים (כמו הפעלת טרנספורמציות או רדוקציה לבעית חיזוי אחרת) במקום להשקיע מאמץ באימון מודלים מתקדמים יותר.

מה חסר במה שעשינו?

אלגוריתמי למידה קלאסיים

- עצי החלטה – בפרק זה נעשו שימוש מקדים ל-set train כאשר נבחרו תוכנות ספציפיות לבניית עץ ויער ההחלטה. בחירת תוכנות מהוות למעשה השרת העמודות מטבלת train-set. בהמשך לפעולה השרת העמודות הינו רוצים לניסות ולהסיר שורות מה-set train, זההינו, להסיר דוגמאות מקובצת האימון. השרה זו תהווה שלב שימוש מקדים אשר באמצעותו נוכל להסיר דוגמאות "רושות" ובכך לייצור set train טוב יותר אשר יועיל לנו בקבלת תוצאות מדויקות יותר על קבוצת המבחן. נשים לב שטיפול בדוגמאות "רושות" בא לידי ביטוי בгиוזם העץ אך קיימים אלגוריתמים שונים לטיפול בעיה זו טרם בנית המודל של עץ ההחלטה או יער ההחלטה.
- KNN – רأינו כי תוצאות החיזוי שלנו משתפרות משמעותית ככל שה-set test mean שלנו גבוהה יותר. למרות ההבנה כי ההבדל בתוצאות נובע בין השאר מבחירה מטтриקוט המדידה, הינו רוצים להיות מסוגלים לחזות בצורה טובה גם באזורי בהם set mean שלנו נמוך יותר.

אלגוריתמי time series

- בעובודה שלנו בפרויקט זה התמקדו החיזוי גרפ הפונקציה המייצגת את מס' המאומתים היומי, וזאת למרות שעבור אלגוריתמי הלמידה הקלסיים שלנו השתמשו בתכונות רבות אחרות אשר עזרו לנו בחיזוי מס' המאומתים. הינו רוצים להיות מסוגלים להשתמש בתכונות אחרות אשר להן קורלציות גבוההות לתמונה אותה אנחנו מנסים לחזות, ולהשתמש במודלים אשר יאפשרו לחזות את התמונה המבוקשת שלנו בהתבסס על תכונות אחרות (ולא רק בהתבסס על פרדיקטים המבוססים על התמונה אותה אנחנו חוזים).
- שילוב משתנים אקסוגניים – בעובודה שביצענו חלק זה ביצענו חיזוי המבוסס אך ורק על התמונה אותה אנחנו מנסים לחזות. בעיות חיזוי ניתן פעמים רבות לשלב משתנים אקסוגניים למודל החיזוי. משתנה אקסוגני הוא משתנה חיצוני למודל אשר לא תלוי בתמונה אותה אנחנו מנסים לנבא. הוספה של משתנים אלו למודלים שלנו יכולה להעшир את יכולת החיזוי של המודל.
- טיפול עדין יותר בעונתיות – בפרויקט שלנו טיפלו בעונתיות על ידי שימוש במודל SARIMA עם lag העונתי השווה לשבועיים. השימוש במודל העונתי הזה בא לתת מענה לירידה הקבועה במס' הבדיקות המבוצעות בסופי שבוע (וכתוצאה לכך ירידה בכמות הנדבקים). נשים לב שהשימוש ב-*lag* העונתי במודל כופה את השימוש בעונתיות גם על ימות החול בשבוע, שם האפקט העונתי לא בהכרח מורגש. הינו רוצים להיות מסוגלים לטפל באפקט העונתי בצורה יותר מעודנת, ולאחר מכן בהם נכוון להשתמש ב-*lag* העונתי אל מול ימים בהם נוכל להשתמש בגרסאות לינארית שונות להשגת חיזוי מדויק יותר.

כיוונים להמשך מחקר

שימוש הנתונים וייצור תוכנות

- הוספת תכונות – נרצה לבחון הוספת תכונות נוספות ו"מעניינות" ל-dataset שלנו. דוגמאות לתכונות אשר עשויות להיות רלוונטיות:
 - מצאנו טבלאות המכילות דירוג סוציאו אקונומי ברמת היישובים. לדעתנו עמודה המכילה את המדר הסוציאו אקונומי של יישוב יכולה לסייע נוספת לאלגוריתם אשר לא ניתן לבטא באמצעות התכונות הנוכחיות.
 - אחוז המאומתים היומי.
 - מקדם ההדבקה.
- טרנספורמציה לעמודות City_Code כך שבין כל שני יישובים קרובים יהיה מרחק מספרי קטן ובין יישובים רחוקים יהיה גבוה. לדוגמה: ה-city_code של חיפה הוא 4000, ושל נשר שצמודה לחיפה 2500 ושל תל אביב שרחוקה מיפה 5000. אם נעשה טרנספורמציה למזהים הללו כך שהמרחב ביןיהם יבטא את המרחק הפיזי, אז האלגוריתמים המודדים את המרחק האוקלידי (כמו KNN) יעדיפו דוגמאות השיכנות לערים הקרובות לעיר השיכנת לדוגמת המבחן.

אלגוריתמי למידה קלסיים

- עצי החלטה – שימוש באלגוריתמים נוספים לבחירה וסינון תוכנות בלבד RFE.

- עצי החלטה – החלפת תכונות נוספות למעט התכונות אשר הוחלפו בחלק ג' (בפרק ה- Decision Tree). כפי שראינו החלפה מושכלת של תכונות ספציפיות עלולה להביא לשיפור בתוצאות. ניתן כי קיימות תכונות נוספות שהחלפתן בתכונות "חכמתו יותר" יניבו תוצאות מהימנות יותר.
- מימוש מגנון *retrain* בעזרת שימוש באלגוריתם *rolling forecast* כפי שביצעו בפרק ה- Time Series Analysis. מכיוון שאלגוריתמי החיזוי הקלסיים עובדים מעל dataset המכיל מספר רב של תכונות, תחילה retrain עשוי להיות שימושי במיוחד ולשקף את ההתפתחות של התכונות כתלות בזמן (לדוגמה – נתוני ההתחסנות אשר משתנים כתלות בתקופת הזמן).
- מסוג על הבניי משלוב של אלגוריתמי הלמידה הקלסיים (DT ו-KNN) – ניתן לחקור לפי ה-test setמתי אנחנו מקבלים תוצאות חיזוי טובות בערך ומתי תוצאות חיזוי טובות ב-KNN. לאחר מכן, נזכיר "מסוג עלי", שמכיל את שני המסוגים, ולפי התכונות בוחר לאן לשולח את דוגמת המבחן, למסוג ה-DT או למסוג ה-KNN. בכך נוכל להשתמש בכל אלגוריתם רק עבור האזוריים בהם הוא חזק יותר. אופציה נוספת היא לבצע חיזוי בצורה של וועדה – כל דוגמת מבחן תישלח אל מסוג מסוג DT ומסוג KNN, וההחלטה תהיה הממוצע של ערכי החיזוי (או פונקציה אחרת של שני הערכים המתקבלים מכל אחד מהמסוגים).

אלגוריתמי time series

- שימוש ב-VAR (vector auto regression) לביצוע חיזוי המבוסס על מס' משתנים. היינו רוצים לשלב תכונות נוספות על מנת לבדוק את השפעתן על גרע המאומתים. תכונות אותן היינו רוצים להצליב: מקדים הבדיקה, אחוז הבדיקות החשובות, אחוז מוחסנים בכל אחת ממנות החיסון (בהתאם חיזוי של תקופה תאריכים מתאימה).
- שילוב של משתנים אקסוגניים באלגוריתמי החיזוי שלנו. משתנים אקסוגניים מעניינים אותנו ניתן לשלב לדוגמה הם אינדיקטורים לימים סוף השבוע אוימי חמ, אינדיקטורים לתקופות התפרצויות או לתקופות בהן הוטלו סגרים. ישנו אלגוריתמי למידה אשר מותאמים לשילוב משתנים אקסוגניים (לדוגמה – אלגוריתם SARIMAX).
- בנייה מסוג משולב הכול בתוכו מס' אלגוריתמי time series שונים. ניתן לבצע אנהיזה על מנת להבין איזה אלגוריתם מתאים לחיזוי עבור אזורים שונים בפונקציית התכונה שלנו. לדוגמה: נניח שאלגוריתם ARMA עבור פרמטרים מסוימים נותן את ביצועי החיזוי הטובים ביותר עבור תקופות רגעה (בזמן אין עליה דרסטית בתחילתה), ומנגד אלגוריתם SARIMA נותן ביצועי חיזוי טובים יותר עבור תקופות התפרצויות. נוכל אם כך לבנות מסוג משולב אשר בהינתן יום לחיזוי בודק תכונות אקסוגניות מסוימות ו/או את ערכי הימים הקודמים, ובהתאם לכך מחזיר תשובה לפי האלגוריתם הרלוונטי.
- וועדה כפתרון לכונון פרמטרים – ראיינו בפרק ההדגמות והניסויים כי ביצוע כוונון פרמטר עבור אלגוריתמי time series עלול להיות לעיתים שימושית לשיטה (בשל ריבוי הפרמטרים השונים). פתרון לבעה זו יכול התקבל על ידי ביצוע כוונון חלקית של פרמטרים בשילוב עם בחירת פרמטרים המבוססת על הכללים אותם ראיינו במהלך הפרק (לדוגמה: ניתוח קורלציות מגף ה-ACF וה-PACF, ניתוח עונתיות בעזרת *seasonal decomposition* ועוד). נוכל לנצל שלוש גרסאות עבור מודל SARIMA, כאשר לכל גרסה פרמטרים שונים. נוכל לאמן את שלושת המודלים ולבצע חיזוי בשיטת וועדה. ערך החיזוי עבור כל יום יתקבל על ידי לקיחת הפעלת פונקציה כלשהי על שלושת

תוצאות החיזוי המתקבלות מהמודלים השונים שאימנו. פונקציה זו יכולה להיות פונקציית **מומוצע/חציון או אפילו בחירה אקראית**.

הוראות הרצתה

- בתיקיית הקוד ובדף GitHub של הפרויקט כבר מופיעים כל ה-datasets הדרושים על מנת להריץ את הקוד. אלו הקבצים שעברו את תהליך preprocessing כדי שהוסבר בפרק ה-Data. בתיקייה Preprocess מופיעים כל קבצי הקוד אשר אחראים על ייצור ה-datasets שלנו. תהליך ייצור ה-datasets משתמש בטבלאות הגולמיות של הנתונים אשר נמצאות בתיקייה Resources. על מנת להריץ באופן ידני את ייצור הטבלאות ניתן להריץ את פונקציית ה-main בקובץ utils.py (פונקציה זו מבצעת עיבוד ראשוני לקבצי הנתונים הגולמיים), ולאחר מכן להריץ את הקובץ create_main_df.py. קובץ זה אחראי לייצור את כל התוכנות עבור ה-dataset שלנו וליצא קובץ csv בשם corona_df.csv אשר נשמר תחת התיקייה Preprocess. **נציין שוב כי אין חובה להריץ שוב את תהליך ייצור הטבלאות על מנת להריץ את הקוד!**

כל קבצי הקוד עברו אלגוריתמי הלמידה השונים שמורים תחת התיקייה **Algorithms**. הוראות להרצת הקוד עברו כל אחד מהאלגוריתמים השונים:

- עצי החלטה - בתיקייה DT יש להריץ את הקובץ DT.py. הרצת קובץ זה תניב את תוצאות הדוח'ה (עבור הפרק של עצי ההחלטה) בזה אחר זה לפי סדר. יש לשים לב שהניסויים לבחירת התוכנות הטובות ביותר ביותר עלולים לרוץ זמן רב, זאת לאור השימוש באלגוריתם RFE אשר בירצטו בונה מספר רב של גרגסורים עד לקבלת התוכנות הרלוונטיות ביותר (חלקים ב' וג'). מצורפות הערות בקוד להפרדה בין החלקים השונים. בנוסף ניתן להריץ את כל אחד מחלקים בנפרד (לא תלות בחלקים האחרים).
- KNN – בתיקייה KNN יש להריץ את הקובץ KNN.py. פונקציית ה-main מכילה קטע קוד ראשון לא בהערה שדרוש לכל הניסויים. כדי להריץ את הניסויים, יש להוריד מהערה כל פעם חלק אחר (כל חלק עטוף בסימני '#'). כל חלק ניתן להריץ בנפרד (לא תלות בחלקים האחרים).
- אלגוריתמי time series – בתיקייה time_series יש להריץ את הקובץ TimeSeriesAnalysis.py. פונקציית ה-main של קובץ זה מರיצה את כל הדוגמאות והניסויים שהוצעו בפרק TimeSeriesAnalysis בזה אחר זה לפי סדר. ניתן גם לבחור להריץ כל פונקציה בנפרד על מנת לקבל את התוצאות עבור חלק ספציפי. כל קבצי ה-time series data הדרושים עבור ההרצאות בחלק זה כבר מופיעים בתיקיית הקוד וב-GitHub. הניסוי הראשוני מרץ הדוגמה של ספריית הטרנספורמציות שלנו (כפי שהציגו בתת הפרק Time Series Transformations), ולאחריו כל הדוגמאות והניסויים העוסקים באלגוריתמי החיזוי השונים. הניסויים רצים בהתאם לסדר בו הם מופיעים בדוח'ה הפרויקט.

ביבליוגרפיה

כללי

ידע מקדים עליו התבססו בפרויקט זה נלקח משקפי ההרצאות והתרגולים של קורס "מבוא לבינה מלאכותית" במוסד הטכניון בהנחיית פרופסור שאול מרקוביץ'. בנוסף נעשה שימוש בצלומים מתוך השקפים.

נמנה את רשימת המקורות שלנו עבור החלקים השונים בפרויקט. מקורות אלה כוללים מקורות ספרותיים וכן גם מקורות מהם נלקחו צילומים ואיורים בהם נעשה שימוש בדוחת.

תמונה לדף שער :

- <https://www.who.int/>

איסוף נתונים :

- <https://data.gov.il/dataset/covid-19>
- <https://datadashboard.health.gov.il/COVID-19/>
- [https://www.cbs.gov.il/he/mediarelease/pages/2020/%D7%90%D7%A4%D7%99%D7%95%D7%9F-%D7%99%D7%97%D7%99%D7%93%D7%95%D7%AA-%D7%92%D7%90%D7%95%D7%92%D7%A8%D7%A4%D7%99%D7%95%D7%AA-%D7%95%D7%A1%D7%99%D7%95%D7%92%D7%9F-%D7%9C%D7%A4%D7%99-%D7%94%D7%A8%D7%9E%D7%94-%D7%94%D7%97%D7%91%D7%A8%D7%AA%D7%99%D7%AA-%D7%9B%D7%9C%D7%9B%D7%9C%D7%99%D7%AA-%D7%A9%D7%9C-%D7%94%D7%90%D7%95%D7%9B%D7%9C%D7%95%D7%A1%D7%99%D7%99%D7%94-2017.aspx](https://www.cbs.gov.il/he/mediarelease/pages/2020/%D7%90%D7%A4%D7%99%D7%95%D7%9F-%D7%99%D7%97%D7%99%D7%93%D7%95%D7%AA-%D7%92%D7%90%D7%95%D7%92%D7%A8%D7%A4%D7%99%D7%95%D7%AA-%D7%95%D7%A1%D7%99%D7%95%D7%95%D7%92%D7%9F-%D7%9C%D7%A4%D7%99-%D7%94%D7%A8%D7%9E%D7%94-%D7%94%D7%97%D7%91%D7%A8%D7%AA%D7%99%D7%AA-%D7%9B%D7%9C%D7%9B%D7%9C%D7%99%D7%AA-%D7%A9%D7%9C-%D7%94%D7%90%D7%95%D7%9B%D7%9C%D7%95%D7%A1%D7%99%D7%99%D7%94-2017.aspx)
- https://data.gov.il/dataset/residents_in_israel_by_communities_and_age_groups/resource/64edd0ee-3d5d-43ce-8562-c336c24dbc1f

מטריקות מדידה :

- https://scikit-learn.org/stable/modules/model_evaluation
- <https://joydeep31415.medium.com/common-metrics-for-time-series-analysis-f3ca4b29fe42>
- https://en.wikipedia.org/wiki/Coefficient_of_determination

עצי החלטה :

- <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>

- <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>
- <https://machinelearningmastery.com/rfe-feature-selection-in-python/>

: KNN

- <https://scikit-learn.org/stable/modules/neighbors.html#neighbors>
- https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
- <https://he.wikipedia.org/wiki/%D7%A8%D7%92%D7%A8%D7%A1%D7%99%D7%94%D7%90%D7%A0%D7%9C%D7%99%D7%96%D7%94>
- <https://www.kaggle.com/tomwarrens/timeseriessplit-how-to-use-it>
- <https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4>
- <https://www.mathworks.com/matlabcentral/fileexchange/63621-knn-classifier>

: אלגוריתמי time series

- <https://vitalflux.com/autoregressive-ar-models-with-python-examples/>
- <https://machinelearningmastery.com/autoregression-models-time-series-forecasting-python/>
- https://scikit-learn.org/stable/modules/model_evaluation
- <https://alkaline-ml.com/pmdarima/modules/classes.html>
- https://en.wikipedia.org/wiki/Akaike_information_criterion
- <https://otexts.com/fpp2/>
- <https://www.statsmodels.org/dev/index.html>
- <https://online.stat.psu.edu/stat510/lesson/1/1.2>
- <https://www.machinelearningplus.com/time-series/time-series-analysis-python/>
- <https://towardsdatascience.com/the-complete-guide-to-time-series-analysis-and-forecasting-70d476bfe775>
- <https://www.baeldung.com/cs/acf-pacf-plots-arma-modeling>
- https://en.wikipedia.org/wiki/Statistical_hypothesis_testing
- <https://www.investopedia.com/>